

(19)日本国特許庁 (J P)

(12) 公表特許公報 (A)

(11)特許出願公表番号

特表2002-520748

(P2002-520748A)

(43)公表日 平成14年7月9日(2002.7.9)

(51)Int.Cl. <sup>7</sup>	識別記号	F I	テーマコード <sup>*</sup> (参考)
G 0 6 T 15/00	2 0 0	G 0 6 T 15/00	2 0 0 5 B 0 5 7
1/20		1/20	C 5 B 0 8 0

審査請求 未請求 予備審査請求 有 (全164頁)

(21)出願番号 特願2000-560548(P2000-560548)  
(86)(22)出願日 平成11年7月16日(1999.7.16)  
(85)翻訳文提出日 平成13年1月12日(2001.1.12)  
(86)国際出願番号 P C T / U S 9 9 / 1 6 0 3 8  
(87)国際公開番号 W O 0 0 / 0 4 5 0 5  
(87)国際公開日 平成12年1月27日(2000.1.27)  
(31)優先権主張番号 6 0 / 0 9 2 , 9 7 7  
(32)優先日 平成10年7月16日(1998.7.16)  
(33)優先権主張国 米国 (U S)

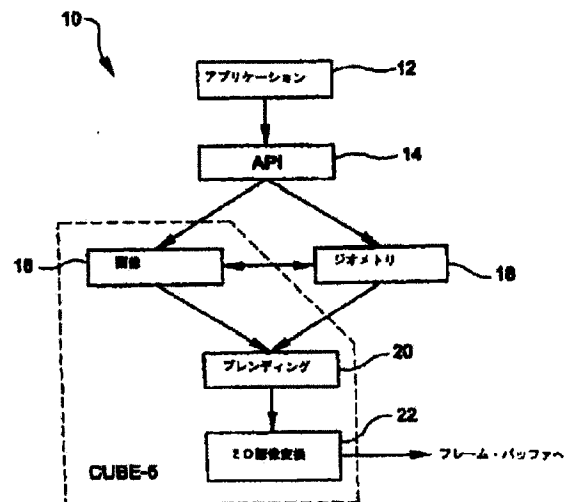
(71)出願人 ザ リサーチ ファウンデーション オブ  
ステイト ユニヴァーシティ オブ ニ  
ューヨーク  
アメリカ合衆国 ニューヨーク州 12201  
-0009 アルパニ ビーオーボックス 9  
(72)発明者 カウフマン アリー イー  
アメリカ合衆国 ニューヨーク州 11803  
ブレインヴィュー セダー ドライヴ  
ウエスト 94  
(74)代理人 弁理士 中村 稔 (外9名)

最終頁に続く

(54)【発明の名称】 リアルタイム・ボリウム処理およびユニバーサル3Dレンダリングのための装置および方法

(57)【要約】

リアルタイム・ボリウム処理およびユニバーサル三次元レンダリング(10)のための装置および方法である。この装置は、複数の三次元メモリ・ユニットと、大域水平方向連絡を与える少なくとも1つのピクセル・バスと、複数のレンダリング・パイプラインと、少なくとも1つのジオメトリ・バスと、制御ユニットとを包含する。複数のレンダリング・パイプラインは、各々、好ましくは、補間、シェーディング、FIFOバッファリング、連絡およびルックアップ表のためのハードウェアを含む。本発明の装置は、表面、画像およびボリウムを単一の画像に混合するためのジオメトリ・パイプライン(18)に接続してあってもよい。3Dボリウムのボリウムメトリック・レイ・キャストリングを実施する方法は、所定の視点から主要投影軸線に沿って距離を計算する段階と、ボリウムを指数的に増大する境界を有する複数の連続領域に分割する段階と、ボリウムを通して視点から複数のレイをキャストリングする段階と、領域境界で2つまたはそれ以上のレイを合併するか、1つまたはそれ以上のレイを分割する段階と、ボリウム全体



**【特許請求の範囲】**

【請求項1】 リアルタイム・ボリューム処理およびユニバーサル三次元（3D）レンダリングのための装置であって、視点を定める視野・処理パラメータに応答するようになっており、また、前記視点から3Dボリューム投影画像または新しい修正ボリュームを生成し、この画像が複数のピクセルを含んでいる装置において、

複数の個別のボクセルを記憶する複数の三次元（3D）メモリ・ユニットであり、前記ボクセルの各々が、それと組み合わせた場所およびボクセル・データを有し、前記ボクセルが一緒になってボリューム・データセットを形成しており、前記視野・処理パラメータが、前記ボリューム・データセットの少なくとも1つのベースプレーンおよび前記ボリューム・データセットの最初と最後の処理スライスを定める三次元メモリ・ユニットと、

大域水平方向連絡を与える少なくとも第1のピクセル・バスと、

複数のレンダリング・パイプラインであって、各々が、前記複数の3Dメモリ・ユニットのうちの対応するものと前記少なくとも第1のピクセルの両方に垂直方向で接続されており、各々が、隣接のパイプラインとの水平方向連絡を有し、前記対応するメモリ・ユニットから前記ボクセル・データを受け取り、前記ボリューム・データセットの面と整合した二次元（2D）ベースプレーン画像を生成するレンダリング・パイプラインと、

前記複数のレンダリング・パイプラインとジオメトリ・エンジンとの間の大域水平方向連絡を与える少なくとも1つのジオメトリ・バスであり、単一画像においてジオメトリック、ボリュメトリックのオブジェクトと一緒にレンダリングするのを可能にする少なくとも1つのジオメトリ・バスと、

前記第1処理スライスを最初にサンプル・ポイントの現スライスとして指定し、前記最後の処理スライスに達するまで前記ボリューム・データセットの以降のスライスを通して現スライスとして走査することを制御する制御ユニットと、  
を包含することを特徴とする装置。

【請求項2】 請求項1の装置において、さらに、前記複数の3Dメモリ・ユニットおよび前記複数のレンダリング・パイプラインに作動可能に接続した

フィードバック・バスを包含し、このフィードバック・バスが、前記レンダリング・パイプラインの出力部と、前記複数のレンダリング・パイプラインの入力部および前記複数の対応する3Dメモリ・ユニットのうちの少なくとも1つとの間の連絡を行うことを特徴とする装置。

【請求項3】 請求項2の装置において、さらに、前記複数の3Dメモリ・ユニットに作動可能に接続してあり、前記3Dメモリ・ユニット間の大域水平方向連絡を与える画像入力バスを包含することを特徴とする装置。

【請求項4】 請求項3の装置において、さらに、前記二次元(2D)ベースプレーン画像を記憶するための複数の二次元(2D)メモリ・ユニットを包含し、これら2Dメモリ・ユニットの各々が、前記複数のレンダリング・パイプラインの対応するものと前記少なくとも1つの第1ピクセル・バスの両方に接続していることを特徴とする装置。

【請求項5】 請求項4の装置において、さらに、前記少なくとも第1ピクセル・バスに作動可能に接続した少なくとも1つのワープ単位を包含し、この少なくとも1つのワープ単位が、所定の画像プレーン上へ前記ベースプレーン画像を組み立て、変換することを特徴とする装置。

【請求項6】 請求項1の装置において、前記複数のレンダリング・パイプラインの各々が、前記対応する1つに作動可能に接続した入力部と1つの出力部を有する少なくとも第1スライス・ユニットであり、前記3Dメモリ・ユニットからボクセル値を受け取る少なくとも第1スライス・ユニットと、

前記少なくとも第1スライス・ユニットの前記出力部に作動可能に接続した入力部と、前記少なくとも第1ピクセル・バスに作動可能に接続した1つの出力部とを有する合成ユニットと、

前記合成ユニットに作動可能に接続した入力部と1つの出力部とを有し、複数のピクセルを記憶し、これらのピクセルが、カラーおよびそれに組み合わせた不透明度を有する合成バッファと、

を包含することを特徴とする装置。

【請求項7】 請求項6の装置において、さらに、少なくとも第2、第3のスライス・ユニットを包含し、第1、第2、第3のスライス・ユニットが、前

記複数の3Dメモリ・ユニットから読み込んだ前記ボクセルを緩衝し、前記ボクセルの引き続く再読み込みを前記スライス・ユニットから実施することを特徴とする装置。

【請求項8】 請求項7の装置において、前記制御ユニットが、ブロック制御・介在要領で前記処理スライスの1横断分を組織化し、それによって、時間的に隣接したサンプル・ポイントが、主要投影軸線に沿った1つのユニット分異なることを特徴とする装置。

【請求項9】 請求項6の装置において、前記レンダリング・パイプラインの各々が、さらに、

前記合成バッファの前記出力部および前記合成ユニットに作動可能に接続したバイリニア補間ユニットと、

前記少なくとも第1スライス・ユニットの前記出力部に作動可能に接続した入力部と、1つの出力部を有するトリリニア補間ユニットと、

前記少なくとも第1スライス・ユニットの前記出力部に作動可能に接続した第1入力部および前記トリリニア補間ユニットの前記出力部に作動可能に接続した第2入力部を有し、そして、1つの出力部を有する勾配計算ユニットと、

この勾配計算ユニットの前記出力部に作動可能に接続した第1入力部および前記トリリニア補間ユニットの前記出力部に作動可能に接続した第2入力部を有し、また、前記合成ユニットの前記入力部に作動可能に接続した出力部を有するシェーディング・ユニットと

を包含することを特徴とする装置。

【請求項10】 請求項9の装置において、前記複数のレンダリング・パイプラインの各々が、さらに、前記トリリニア補間ユニットおよび前記勾配計算ユニットを前記バイリニア補間ユニットに作動可能に接続するテクスチャ・マップ・バイパス・バスを包含することを特徴とする装置。

【請求項11】 請求項9の装置において、前記複数のレンダリング・パイプラインの各々が、さらに、前記少なくとも1つのピクセル・バスと前記複数の3Dメモリ・ユニットの前記対応する1つと間の少なくとも1つのフィードバック・コネクションを包含し、前記フィードバック・コネクションが、前記少な

くとも1つのピクセル・バスから前記3Dメモリ・ユニットへのフィードバックを行い、次いで、前記複数のレンダリング・パイプラインにおける任意の中間ステージまでフィードバックを行うように構成してあることを特徴とする装置。

【請求項12】 請求項9の装置において、前記勾配計算ユニットが、乗算器と、加算器とを包含し、

各主要投影軸線に沿って、3つの連続したサンプル・ポイントの加重平均を計算し、前記レンダリング・パイプラインおよび前記対応する3Dメモリ・ユニットに作動可能に接続したフィードバック・コネクションを通じて書き戻すことを特徴とする装置。

【請求項13】 請求項1の装置において、前記制御ユニットが、前記ボリューム・データセットのビュー・フラスタムを、領域境界によって分離された複数の領域へ分割し、前記領域が、前記視点に最も近い第1領域を包含し、各連続した領域が、前記第1の領域で開始し、その前の隣接した領域のその2倍である、主要投影軸線に沿ったデプスを有することを特徴とする装置。

【請求項14】 請求項13の装置において、複数の連続したサイト・レイが、前記視点から前記ボリューム・データセットを通してキャストされ、前記複数のサイト・レイが、前記ボリューム・データセットを通過するときに互いに関し、発散し、前記制御ユニットが、

(i) 前記サイト・レイのうちの少なくとも2つを結合すること、

(ii) 前記サイト・レイの少なくとも1つを分割すること

のうち一方を可能にすることを特徴とする装置。

【請求項15】 三次元(3D)ボリューム・データセットの透視ボリュームメトリック・レイ・キャストを実施する方法であって、前記ボリューム・データセットが、複数のボクセル・グリッド・ポイント場所を包含し、また、ビュー方向に対してほとんど直角な前記ボリューム・データセットの面を構成するベースプレーンを有する方法において、複数の三次元(3D)メモリ・ユニットに分散して記憶した個別のボクセルを利用し、前記ボクセルの各々が、それと組み合わせた場所およびボクセル・データを包含し、さらに、

(a) 前記視点からの前記ボリューム・データセットの前記ベースプレーンまでの主要投影軸線に沿った距離を計算する段階と、

(b) 前記ボリューム・データセットを領域境界によって分離された複数の連続領域へ分割する段階であり、前記領域境界が、前記主要投影軸線に沿った前記視点からの指数的に増加する距離のところに位置しており、前記複数の領域が、少なくとも、ボリューム・データセットを含んでおり、前記第1領域後の各引き続く領域が、先の隣接した領域のデプスの2倍である、前記主要投影軸線に沿ったデプスを有し、

(c) 前記視点から前記ボリューム・データセットを通して延びる複数の連続したサイト・レイをキャストし、前記サイト・レイが、現サンプル領域を通過する段階と、

(d) 以下の1つを実施する段階であり、すなわち、

(i) 前記サイト・レイのうちの少なくとも2つを結合すること、および、

(ii) 前記サイト・レイのうち少なくとも1つを分割することのうち1つを実施する段階と、

(e) 前記最終処理スライスに到達するまで前記ボリューム・データセットの、前記第1処理スライスで開始する引き続く領域を順次にサンプリングし、前記引き続く領域の各々が、順次に、現サンプル領域となるように段階(c)、(d)を繰り返す段階のうち1つを実施する段階とを包含することを特徴とする方法。

【請求項16】 請求項15の方法の透視レイ・キャスト方法において、段階(c)において、前記複数のサイト・レイが、前記ボリューム・データセットを通してキャストされ、前記サイト・レイの各々が、前記領域境界のところで前記ボリューム・データセットにおける前記ボクセル・グリッド・ポイント場所と一致することを特徴とする透視レイ・キャスト方法。

【請求項17】 請求項15の透視レイ・キャスト方法において、段階(d)において、前記結合段階を、所定のフィルタ重みを有するフィルタで実施し、前記分割を、リニア補間で実施することを特徴とする透視レイ・キャスト方法。

【請求項18】 請求項17の透視レイ・キャスト方法において、

前記フィルタが、パートレット・フィルタであることを特徴とする透視レイ・キャストリング方法。

【請求項19】 請求項15の透視レイ・キャストリング方法において、段階(d)において、複数の前記サイト・レイが、新しいサイト・レイを生成するように結合され、この新しいサイト・レイが、前記結合したサイト・レイの中心サイト・レイの正確な連続体であることを特徴とする透視レイ・キャストリング方法。

【請求項20】 ソース・ボリューム・データセットのディテール・レベル(LOD)表現を生成する方法であって、前記ボリューム・データセットが、複数の三次元(3D)メモリ・ユニットに分散して記憶された複数の個別のボクセルを包含し、前記ボクセルの各々が、それと組み合わせた場所およびボクセル・データを有する方法において、

(a) 複数の新サンプルを生成するように前記ボリューム・データセットを再サンプリングする段階と、

(b) 前記複数の新サンプルのうち他のサンプルすべてを廃棄し、前記ソース・ボリューム・データセットよりサイズがコンパクトである第2ボリューム・データセットを形成する段階と、

(c) 前記第2ボリューム・データセットのスライスを前記3Dメモリ・ユニットに書き込み、それによって、前記ソース・ボリューム・データセットを前記第2ボリューム・データセットに置き換える段階と、

(d) 第2ボリューム・データセットが所定のサイズになるまで段階(a)～(c)を繰り返す段階とを包含することを特徴とする方法。

【請求項21】 請求項20の方法において、段階(a)の再サンプリングをリニア補間を使用して実施することを特徴とする方法。

【請求項22】 請求項20の方法において、段階(a)の再サンプリングを、x、yおよびz方向において0.5にセットした重みをもってトリリニア補間によって実施することを特徴とする方法。

【請求項23】 ボリューム・データセットのディテール・レベル(LOD)表現をレンダリングする方法であって、前記ボリューム・データセットが複

数の三次元（3D）メモリ・ユニットに分散して記憶された複数の個別のボクセルを包含し、各ボクセルが、前記ボリューム・データセット内のグリッド・ポイントに位置する場所を有し、また、それと組み合ったボクセル・データを有する方法において、

（a）視点および出発ディテール・レベル（LOD）を定める視野・処理パラメータを選択する段階であり、前記出発LODがスクリーン・スペース画像サイズにほぼ等しいベースプレーン画像を生成し、前記ベースプレーン画像が、視野方向に対して最も垂直である前記ボリューム・データセットの面を定めている段階と、

（b）前記ボリューム・データセットを複数のスラブに分割し、各スラブが、前記視野方向に沿って、前記スラブの前後における投影ボクセル距離が2の因数だけ異なるようにする厚さを有し、前記複数のスラブが、前記ボリューム・データセットの最も詳細な表現の出発スラブを含んでいる段階と、

（c）前記出発スラブを合成バッファ内にレンダリングし、この合成バッファが、前記描写スラブを表している合成バッファ画像を記憶する段階と、

（d）前記合成バッファ画像を0.5の因数だけコンパクト化する段階と、

（e）前記縮尺した合成バッファ画像を前記合成バッファへ書き込む段階と、

（f）前記ボリューム・データセットのすべてのスラブが処理されてしまうまでボリューム・データセットの前記複数のスラブの各引き続くスラブを半解像度が順次に読み込むことによって段階（c）、（d）を繰り返す段階とを包含することを特徴とする方法。

【請求項24】 請求項23の方法において、さらに、ワーブ単位を用意する段階を包含し、段階（d）を前記ワーブ単位によって実施することを特徴とする方法。

【請求項25】 複数のソース・ピクセルを包含しているソース画像および複数の目標ピクセルを包含している目標画像との間でジオメトリック変換を実施する順方向ワーピング方法であって、前記ソース画像を視点から生成する順方向ワーピング方法において、

前記視点および複数のスキャンラインを定める視野・処理パラメータを選ぶ段



階と、

平行・保存スキャンライン方向を決定する種家であり、前記ソース画像内の平行なスキャンラインが前記目標画像において平行のままとなるようにする段階と

、

前記スキャンラインのすべてがマッピングされるまで前記ソース画像の引き続くスキャンラインを通して増分で走査することによって前記平行・保存スキャンライン方向に沿って前記目標画像に前記ソース画像をマッピングする段階と

を包含することを特徴とする順方向ワーピング方法。

【請求項26】 請求項25の順方向ワーピング方法において、前記平行・保存スキャンライン方向が、透視変換マトリックスに由来することを特徴とする順方向ワーピング方法。

【請求項27】 請求項25の順方向ワーピング方法において、さらに、前記ソース画像を通して前記スキャンラインを走査する段階であり、各スキャンラインに沿ったサンプルを増分計算する段階と、

所定の読み込みテンプレートに従って固定パターンにおける前記ソース・ピクセルを読み込む段階と、

所定の書き込みテンプレートに従って前記ソース・ピクセルのサンプル値を書き込む段階と

を包含することを特徴とする順方向ワーピング方法。

【請求項28】 請求項27の順方向ワーピング方法において、さらに、前記ソース画像における隣接したサンプルまでの距離を算出することによってフットプリントを計算する段階を包含し、前記平行・保存スキャンライン方向に沿ったサンプルが等距離であり、各ソース・ピクセルの最も近い隣接目標ピクセルまでの距離を増分計算することを特徴とする順方向ワーピング方法。

【請求項29】 少なくとも1つのボリュームおよび複数のジオメトリを含んでいるシーンをレンダリングし、画像を生成し、この画像を視点から生成する方法であって、

(a) 複数のジオメトリ・プリミティブを用意し、前記ジオメトリ・プリミティブの各々がそれと組み合った不透明度を有し、また、複数のボクセルを含むボリ

ューム・データセットを用意し、各ボクセルが、それと組み合った場所とボクセル・データを有する段階と、

(b) 前記視点、前記ボリューム・データセットの少なくとも1つのベースプレーンであり、視野方向に対して最も垂直である前記ボリューム・データセットの面を定めるベースプレーンおよび前記ボリューム・データセットの最初と最後の処理スライスを定める視野・処理パラメータを選ぶ段階と、

(c) 前記不透明度に従って、半透明ジオメトリ・プリミティブから不透明ジオメトリ・プリミティブを分離する段階と、

(d) 前記不透明なジオメトリ・プリミティブをZバッファへレンダリングする段階であり、前記Zバッファが、ボリューム・サンプル距離に一致するに十分な解像度を有し、前記視点から前記視野方向に沿った距離を表しているZバッファ画像および前記ジオメトリ・プリミティブの各々についての関連したサンプル値を記憶し、前記Zバッファ画像が、前記ボリューム・データセットの前記第1処理スライスと一致するプレーン上にある段階と、

(e) 合成バッファを、前記不透明なジオメトリ・プリミティブの前記Zバッファ画像で初期化する段階と、

(f) 複数のサイト・レイを前記ボリューム・データセットの前記第1処理スライスを通してキャストする段階と、

(g) 視野方向に沿って採った、前記ボリューム・データセットの現サンプルのデプスが前記Zバッファ画像のデプスより大きいかどうかについて決定する段階と、

(h) 前記ボリューム・サンプルのデプスが前記不透明なジオメトリ・プリミティブのデプスより大きい場合に前記ボリューム・サンプル上に前記不透明なジオメトリ・プリミティブを合成する段階と、

(i) 前記最後の処理スライスに達するまで前記ボリューム・データセットの引き続くスライスを通して段階(f)～(h)を繰り返す段階と

を包含することを特徴とする方法。

【請求項30】 請求項29の方法において、さらに、前記半透明のジオメトリ・プリミティブを複数の薄いスラブにレンダリングし、各スラブが、すべ

てのスラブの合併が前記ボリューム・データセットの任意領域を失ったり、重複したりしないように選んだ境界を有するようにする段階と、

前記最後の処理スライスに達するまで前記ボリューム・データセットの2つの隣接したスライス間に前記半透明のジオメトリ・プリミティブの前記薄いスラブを合成する段階と

を包含することを特徴とする方法。

【請求項31】 請求項30の方法において、前記ボリューム・スライスおよび前記半透明のポリゴン・スラブからのデータを別の方法で相互にダブテーブルすることを特徴とする方法。

【請求項32】 請求項30の方法において、半透明のジオメトリ・プリミティブをレンダリングする段階が、複数のバケットを創作し、各バケットが、前記ボリューム・データセットの2つの隣接したスライス間の薄いスラブに対応するようにする段階と、シーン内ですべての半透明のジオメトリ・プリミティブを移動させ、各半透明のジオメトリ・プリミティブを、前記半透明のジオメトリ・プリミティブが交差する各スラブのためのバケットに内に置く段階とを包含することを特徴とする方法。

【請求項33】 請求項30の方法において、半透明のジオメトリ・プリミティブをレンダリングする段階が、事前描写した半透明のジオメトリ・プリミティブの能動ディスプレイ・リストを維持する段階を包含し、それによって、半透明のジオメトリ・プリミティブの各々が、半透明のジオメトリ・プリミティブが交差する前記ボリューム・データセットの第1スライスで前記能動リスト内に置かれ、半透明のジオメトリ・プリミティブがもはやボリューム・データセットの任意のスライスと交差しなくなったときに前記能動リストから除かれることを特徴とする方法。

【請求項34】 所定の視点に関してボリューム・データセットの任意の三次元回転を実施する方法であって、前記ボリューム・データセットが、複数の三次元メモリ・ユニット内に分散して記憶された複数の個別のボクセルを包含し、各ボクセルが、前記ボリューム・データセットのグリッド・ポイント上の場所を有し、それと組み合ったボクセル・データを有する方法において、

前記視点に関して前記ボリューム・データセットの最終的な向きを定める目標三次元回転マトリックスを受け取り、この回転マトリックスを、x、y、z軸線に沿った軸線回転の連結として表す段階と、

前記回転マトリックスを前記ボリューム・データセットの連続的シャワー変換のうち少なくとも2つのパスからなるシーケンスに分解する段階と  
を包含することを特徴とする方法。

【請求項35】 請求項34の三次元回転方法において、分解段階が、以下のシーケンスでの連続的二次元(2D)シャワー変換の4つのパスを包含し、このシーケンスが、

第1の主要方向軸線に沿ってボリューム・データセットの第1の2Dビーム・シャワー変換を実施すること、

第2の主要方向軸線に沿ってボリューム・データセットの第2の2Dビーム・シャワー変換を実施すること、

第3の主要方向軸線に沿ってボリューム・データセットの第3の2Dビーム・シャワー変換を実施すること、そして、

前記第1の主要方向軸線に沿ってボリューム・データセットの第4の2Dビーム・シャワー変換を実施すること

からなることを特徴とする三次元回転方法。

【請求項36】 請求項34の三次元回転方法において、分解段階が、以下のシーケンスでの連続的二次元(2D)シャワー変換の4つのパスを包含し、このシーケンスが、

第1の主要方向軸線に沿ってボリューム・データセットの第1の2Dスライス・シャワー変換を実施すること、

第2の主要方向軸線に沿ってボリューム・データセットの第2の2Dスライス・シャワー変換を実施すること、

第3の主要方向軸線に沿ってボリューム・データセットの第3の2Dスライス・シャワー変換を実施すること、そして、

前記第1の主要方向軸線に沿ってボリューム・データセットの第4の2Dスライス・シャワー変換を実施すること

からなることを特徴とする三次元回転方法。

【請求項37】 請求項36の三次元回転方法において、分解段階が、連続的三次元（3D）ビーム・シャー変換の2つのパスを包含し、第1のパスが、前記第1、第2の2Dスライス・シャー変換の連結であり、第2のパスが、前記第3、第4の2Dスライス・シャー変換の連結であることを特徴とする三次元回転方法。

【請求項38】 請求項34の三次元回転方法において、分解段階が、以下のシーケンスでの連続的二次元（2D）シャー変換の3つのパスを包含し、このシーケンスが、

ボリューム・データセットの第1の2Dビーム・スライス・シャー変換を実施することを包含し、そこにおいて、前記シャー変換が、各々異なった腫瘍軸線方向に沿った、1つのビーム・シャー変換と1つのスライス・シャー変換の積であり、また、

ボリューム・データセットの第2の2Dビーム・スライス・シャー積変換を実施することを包含し、そこにおいて、前記シャー変換が、各々異なった主要軸線に沿った、1つのビーム・シャー変換と1つのスライス・シャー変換の積であり、さらにまた、

前記第1の2Dビーム・スライス・シャー変換のビーム・シャーと同じ主要軸線方向に沿ってボリューム・データセットの2Dビーム・シャー変換を実施すること

を包含することを特徴とする三次元回転方法。

【請求項39】 請求項34の三次元回転方法において、さらに、複数のバレル・シフトを提供する段階を包含し、前記分解段階の前記シャー変換が前記バレル・シフトによって実施されることを特徴とする三次元回転方法。

【請求項40】 請求項34の三次元回転方法において、さらに、複数の対数シフトを提供する段階を包含し、前記分解段階の前記シャー変換が前記対数シフトによって実施されることを特徴とする三次元回転方法。

【請求項41】 請求項34の三次元回転方法において、さらに、

(a) 前記ソース・ボリューム・データセットのソース・ボクセルと目標ボリュ

ーム・データセットの目標ボクセルとの間に1対1の対応関係をセットアップする段階と、

(b) 前記ソース・ボクセルの各々について、前記1対1の対応関係を使用して前記対応する目標ボクセルを研鑽する段階と、

(c) 再サンプリングによってソース・ボリューム・データセットにおける前記ソース・ボクセル場所に前記目標ボクセルの各々を記憶する段階と、

(d) 前記目標ボクセルを前記目標ボリューム・データセットにおける対応する宛先へ変換する段階と

を包含することを特徴とする三次元回転方法。

【請求項42】 請求項41の三次元回転方法において、

段階(c)における前記再サンプリングを、局所的ボクセルへの補間によって実施し、前記目標ボクセルの各々のサンプリング位置を、前記回転マトリックスの逆方向変換を使用して計算し、

段階(d)における前記変換を最も近い隣接の実装機構を用いて実施することを特徴とする三次元回転方法。

【請求項43】 複数のボリュームを含むシーンをレンダリングし、そして、1つの画像を視点から生成する方法において、

(a) 第1のボリューム・データセットと、この第1のボリューム・データセットに対応する第1の変換マトリックスを用意する段階と、

(b) 第2のボリューム・データセットと、この第2ボリューム・データセットに対応する第2の変換マトリックスを用意する段階と、

(c) 前記視点および前記第1、第2のボリューム・データセットの最初と最後の処理スライスを定める視野・処理パラメータを選ぶ段階と、

(d) それぞれ、前記第1、第2変換マトリックスに従って前記第1、第2ボリューム・データセットを変換する段階と、

(e) 前記第1のボリューム・データセットの前記最初の処理スライスを通して前記視点から視野方向に沿った複数のサイト・レイをキャストする段階と、

、

(f) 前記第2ボリューム・データセットの前記最初の処理スライスを通して前

記視点から前記視野方向に沿って複数のサイト・レイをキャストする段階と、

(g) 前記第1のボリュームの前記第1のスライスを合成バッファへ書き込む段階と、

(h) 前記第2ボリュームの前記最初のスライスを前記合成バッファ上に合成する段階と、

(i) 前記第1、第2のボリューム・データセットの前記最後の処理スライスに達するまで前記視野方向に沿って前記第1、第2のボリューム・データセットの引き続くスライスを通して順次に走査することによって段階(e)～(h)を繰り返す段階と

を包含することを特徴とする方法。

【請求項44】 請求項43の方法において、段階(d)が、さらに、前記第1、第2の変換マトリックス間の差を計算し、前記第1、第2のボリューム・データセットの一方のみを前記差に従って変換する段階を包含することを特徴とする方法。

【請求項45】 請求項44の方法において、段階(d)が、さらに、シャワー変換が前記ボリューム・データセットについて実施された場合に、前記ボリューム・データセットを再サンプリングして補正ボリューム・データセットを形成し、この補正ボリューム・データセットが、前記シャワー変換により導入されたエラーをほぼ除去することを特徴とする方法。

【請求項46】 請求項45の方法において、前記再サンプリング段階を、段階(e)、(f)のうち少なくとも一方での前記レイ・キャスト中に実施することを特徴とする方法。

【請求項47】 複数のボリュームを含むシーンをレンダリングし、視点から1つの画像を生成する方法において、

(a) 第1のボリューム・データセットおよび対応する第1の変換マトリックスおよび第2ボリューム・データセットおよび対応する第2の変換マトリックスを用意し、前記第1、第2のボリューム・データセットの各々が、複数の三次元メモリ・ユニットに分散して記憶された複数の個別のボクセルを包含しており、前

記ボクセルの各々が、前記ボリューム・データセットにおけるグリッド・ポイント上に位置する場所を有し、それと組み合わせたボクセル・データを有する段階と、

(b) 前記視点および前記第1、第2のボリューム・データセットの最初と最後の処理スライスを決める視点・処理パラメータを選択する段階と、

(c) 前記第1、第2のボリューム・データセットの前記最初の処理スライスを通して視野方向に沿って複数のサイト・レイをキャストする段階と、

(d) 前記サイト・レイに沿った各サンプル・ポイントでの前記第1、第2のボリューム・データセットからボクセルの対応する値を新しい値に結合することによって新しいボリューム・データセットを創り出し、前記新しい値の各々が、変調したカラー値を含む段階と、

(e) 段階(c)を繰り返して、前記第1、第2のボリューム・データセットの前記最後の処理スライスにたつするまで前記ボリューム・データセットの引き続くスライスを通して走査することによって段階(c)、(d)を繰り返し、それによって、前記視点に対する単一のボリューム・データセットとして前記新しいボリューム・データセットをレンダリングする段階と

を包含することを特徴とする方法。

【請求項48】 請求項47の方法において、段階(d)が、さらに、前記対応するボクセル値の各々の最大値を前記第1、第2のボリューム・データセットから計算する段階を包含し、ボクセルの結合を、前記最大値を、前記新しいボリューム・データセットにおける対応する新しい値として使用して合成することによって実施することを特徴とする方法。

【請求項49】 請求項47の方法において、段階(d)が、さらに、前記第1、第2のボリューム・データセットからの前記対応するボクセル間の差値を計算する段階を包含し、ボクセルの結合を、前記新しいボリューム・データセットにおける対応する新しい値として前記差値を使用して合成することによって実施することを特徴とする方法。

【請求項50】 請求項47の方法において、段階(d)が、さらに、前記第1、第2のボリューム・データセットからの前記対応するボクセルの各々の



最小値を計算する段階を包含し、ボクセルの結合を、前記新しいボリューム・データセットにおける対応する新しい値として前記最小値を使用して合成することによって実施することを特徴とする方法。

【請求項51】 請求項47の方法において、段階(d)が、さらに、前記第1、第2のボリューム・データセットの各ボクセルに所定のRGB $\alpha$ 値を割り当てることによって前記第1、第2のボリューム・データセットを分類し、シェーディングする段階と、

対応するボクセルの前記RGB $\alpha$ 値を変調してRGB $\alpha$ ボリューム・データセットを形成することによって前記第1、第2のボリューム・データセットの対応するボクセルを結合する段階と

を包含することを特徴とする方法。

【請求項52】 請求項51の方法において、段階(d)が、さらに、前記新しいボリューム・データセットにおける前記ボクセルの各々に対応するインデックス・フィールドを創り出す段階を包含し、前記インデックス・フィールドが、前記新しいボリューム・データセットの発祥形態における各ボクセルをボリューム・データセット化することを示すデータを記憶することを特徴とする方法。

【請求項53】 x勾配成分、y勾配成分およびz勾配成分のうち少なくとも1つを演算する段階を包含する、画像の高品質レンダリングを実施する方法において、

前記x勾配成分が、(i) 重み-1、0、1をもってx方向に加重平均フィルタを適用するサブ段階と、(ii) 重み1、w、1をもってy方向に加重平均フィルタを適用するサブ段階と、(iii) 重み1、w、1をもってz方向に加重平均フィルタを適用するサブ段階とを包含し、

前記y勾配成分が、(i) 重み-1、0、1をもってx方向に加重平均フィルタを適用する段階と、(ii) 重み1、w、1をもってy方向に加重平均フィルタを適用する段階と、(iii) 重み1、w、1をもってz方向に加重平均フィルタを適用する段階とを包含し、

前記z勾配成分が、(i) 重み-1、0、1をもってx方向に加重平均フィル

タを適用する段階と、(ii) 重み1、w、1をもってy方向に加重平均フィルタを適用する段階と、(iii) 重み1、w、1をもってz方向に加重平均フィルタを適用する段階とを包含する

ことを特徴とする方法。

【請求項54】 ジオメトリ・パイプラインに複数のCube-5レンダリング・パイプラインを接続する方法であって、

スクリーン・リフレッシュ・ユニットにデータを供給するに十分なバンド幅を有する複数の外部記憶ユニットを設ける段階を包含し、

前記外部記憶ユニットがフレーム・バッファを形成し、このフレーム・バッファが、ボリューム・レンダリングおよびジオメトリ・レンダリングのうちの1つを実施するためのRGBα、デプスおよびステンシル値のうちの少なくとも1つを記憶し、前記フレーム・バッファが、前記複数の記憶ユニットを横切ってインタリーブされており、Cube-5ボリューム・レンダリング・パイプラインに順番のアクセスで外部記憶ユニットのバンド幅の完全使用を可能にしており、さらに、

前記ジオメトリ・パイプラインが正しく前記フレーム・バッファにアクセスするように前記バッファ・インタリーブ構造の前記ジオメトリ・パイプラインに指示を与える段階を包含することを特徴とする方法。

【請求項55】 ジオメトリ・パイプラインにCube-5ボリューム・レンダリング・パイプラインを接続する方法であって、前記Cube-5レンダリング・パイプラインがフレーム・バッファを包含している方法において、

前記ジオメトリ・パイプラインから前記Cube-5パイプラインにジオメトリ・データを伝達する段階であり、前記ジオメトリ・データがラスターライズされたジオメトリの画像を含んでいる段階と、

前記Cube-5パイプラインにおける前記ジオメトリ・データを結合する段階と、

前記フレーム・バッファ内に前記結合されたジオメトリック・データを記憶する段階と

を包含することを特徴とする方法。

【請求項56】 請求項55の方法において、さらに、実行長コード化（RLE）フォーマットで前記ジオメトリ・データをコード化する段階を包含することを特徴とする方法。

【請求項57】 請求項56の方法において、さらに、実行長コード化ラスタライゼーション・エンジンを設ける段階と、前記フレーム・バッファに前記ラスタライゼーション・エンジンを接続する段階とを包含することを特徴とする方法。

**【発明の詳細な説明】****【0001】****(政府権利声明)**

本発明は、全米科学財団によって与えられた認可MIP9527694の下で、そして、海軍研究局によって与えられた認可N000149710402の下で政府支援により発明されたものである。政府は、この発明について一定の権利を有する。

**【0002】****(発明の背景)**

本出願は、1998年7月16日に出願された合衆国暫定特許出願番号第60／092，977号に対する優先権を主張する。

**【0003】****(発明の分野)**

本発明は、全般的に、三次元(3D)グラフィックスおよびボリューム視覚化に関する。一層詳しくは、本発明は、リアルタイム・ボリューム処理およびユニバーサル三次元レンダリングのための装置および方法に関する。

**【0004】****(従来技術の説明)**

コンピュータ・レンダリングとは、情報の保全性および精度を維持しながら人間の思考にとって理解できるフォーマットに複雑な情報を変換するプロセスである。三次元現象に関する情報からなるボリュメトリック・データは、改良した画像レンダリング技術から利益を得ることができる複雑な情報の1種である。所与の視点からの、ボリュメトリック・データを提示するプロセスは、普通、ボリューム・レンダリングと呼ばれる。

**【0005】**

ボリューム視覚化とは、スーパーコンピュータ・シミュレーションによって、あるいは、ボリューム・グラフィックス技術を使用してジオメトリ模型を合成することによって、入力装置(たとえば、生医学的スキャナ)で生成した大量のボリュメトリック・データを解釈する際に不可欠な技術である。ボリュメトリック・オブジェクトの操作、表示にとって特に重要なのは、映像及びレンダリング・

パラメータの対話的な変更、リアルタイムの表示速度、そして多くの場合、発展しつつある統合入力視覚化システムにおけるような、四次元（4D）視覚化（すなわち、時空的視覚化）と呼ばれる動的なデータセットの経時的变化を見ることが可能なことである。

#### 【0006】

ボリュメトリック・データセットは、普通、ボリューム要素（ボクセル）の3Dグリッドとして表され、しばしば、ボクセルのフル3Dラスタ（すなわちボリューム・バッファ）として記憶される。ボリューム・レンダリングは、ボリューム・データセットのグリッド・ポイントでボクセルによって表される連続的なオブジェクトや現象の3Dスカラー場を視覚化するための、最も普通の技術の一つであって、2つの主要な方法、すなわち、オブジェクト・オーダー（object-order）法および画像・オーダー（image order）法を使用して達成することができる。オブジェクト・オーダー法を使用すると、スクリーン・ピクセルに対する各ボクセルの寄与率が算出され、これらの寄与率を結合して最終的な画像が生成される。画像・オーダー法を使用すると、サイト・レイ（sight ray）が、ボリューム・データセットを通してスクリーン・ピクセルからキャストイングされ、これらのサイト・レイに沿ったボクセルの寄与率を用いて対応するピクセル値を評価する。

#### 【0007】

過去30年にわたって、グラフィックス・システムは、倍々で発展してきた。初期の二次元（2D）から3D、4D（空間と時間）に発展し、ベクトル・グラフィックスからラスタ・グラフィックスへと発展し、ラスタ・グラフィックスでは、基本的グラフィックス・プリミティブとして、ベクトルがポリゴンに置き換えられた。このことは、毎秒数百万個の三角形を表示するのに適合化されたポリゴン・ベースのジオメトリエンジンの普及につながった。しかしながら、このようなシステムでは、三角形のファセットでオブジェクトの形状を近似させているだけである。3Dポリゴン・ベースのグラフィックス市場は、いまだにブームであり続け、パソコン（PC）業界の最も熱い競争の舞台の一つになっている。

#### 【0008】

伝統的なグラフィックス・システムに対する需要が急増しているのに応え、ジオメトリモデルの視覚的な現実感を向上させると共に、オブジェクト形状、構造を向上あるいは交換するために、不連続な画像を処理し表示するための種々の技術が考案されてきた。これら技術としては、2Dテクスチャ・フォト・マッピング、環境マッピング、画像ベース・レンダリング用のレンジ画像、2Dミップ・マッピング、ビデオストリーム、3Dボリューム、3Dミップ・マッピング、4D明視野・ルミグラフおよび五次元(5D)プレノプチック(plenoptic)機能がある。これらすべての技術は、或る種の、不連続なピクセル、テクセル、ボクセルまたはnオクセル間の次元ベース補間(バイリニア、トリリニア、クワッドリニア等)を必要とする。

#### 【0009】

ボリューム視覚化のための特殊な目的のコンピュータ・アーキテクチャおよび方法がこの技術分野では公知である。伝統的なボリューム視覚化方法は、代表的には、オブジェクトの正確な表現を提供するために逐次にボリューム・データセットを走査することによって作動する。たとえば、Dr. Arie Kaufman, Ingmar Bitter and Dr. Hanspeter Pfister (このうちの何人かは本願における発明者として名を連ねている)によって開発されたアーキテクチャ、Cube-4がある。Cube-4は、スライス・パラレル・レイキャスティングに基づく特殊目的スケラブル・ボリューム・レンダリング・アーキテクチャである。Cube-4は、高解像度データセットの真のリアルタイム・レイキャスティング(たとえば、30ヘルツのフレーム速度で $1024^3$ の16ビット・ボクセル)をデリバリーすることができる。しかしながら、Cube-4は、透視投影についてはこのようリアルタイム性能をデリバリーすることはできない。現在のところ、公知の従来技術レンダリング・システムにおいては、透視投影を使用しても、レンダリング時間を増やすか、あるいは、投影画質を低下させるかのいずれかである。さらに、従来のアーキテクチャは、ボリュームおよびジオメトリを単一の画像に結合する能力を与えない。

#### 【0010】

次に図1を参照して、ここには、普通のボリューム視覚化システム1が示して

ある。図1に示すように、ボリューム・データは、ディスク2に記憶されており、レンダリング前にメモリ4にロードされる。次に、中央処理ユニット（CPU）6が、メモリ4内に存在しているデータからボリューム描写画像を計算する。最終的な画像は、フレーム・バッファ8に書き込まれる。このフレーム・バッファは、代表的には、モニタ9または類似のディスプレイ装置上に表示するために、グラフィックス・カードに埋め込まれている。

#### 【0011】

したがって、本発明は、画像データ処理の新世代と考えられ得る程度まで、公知の方法および装置の能力をかなり向上させる方法および装置を提供することを目的とする。

#### 【0012】

他のおよび更なる目的は、本開示の結果として技術者の知るところであり、開示した発明の結果として実現されるすべての目的を含むことを意図している。

#### 【0013】

##### （発明の概要）

本発明は、網羅する新しい特性の故に、従来技術からの新発展に等しい。本発明によれば、リアルタイム・ボリューム処理およびユニバーサル三次元（3D）レンダリングのための装置は、1つまたはそれ以上の三次元（3D）メモリ・ユニットと、少なくとも1つの第1ピクセル・バスと、1つまたはそれ以上のレンダリング・パイプラインと、1つまたはそれ以上のジオメトリ・バスと、制御ユニットとを包含する。この装置は、視点を定める視野・処理パラメータに応答し、視点から3Dボリューム投影画像を生成する。投影画像は、複数のピクセルを包含する。

#### 【0014】

3Dメモリ・ユニットは、複数の個別のボクセルを記憶する。各ボクセルは、それと組み合わせた場所およびボクセル・データを有する。ボクセルは、一緒になって、ボリューム・データセットを形成する。そして、視野・処理パラメータは、ボリューム・データセットのベースプレーンとしてのボリューム・データセットの少なくとも1つの面と、ボリューム・データセットの最初と最後の処理ス

ライスとを定める。制御ユニットは、最初、サンプル・ポイントの現スライスとして最初の処理スライスを示し、最後の処理スライスに達するまで現スライスとしてボリューム・データセットの引き続くスライスを通しての走査を制御する。

#### 【0015】

レンダリング・パイプラインの各々は、垂直方向において、3Dメモリ・ユニットの対応する1つと少なくとも第1ピクセル・バスの両方に接続し、レンダリング・パイプラインの各々は、好ましくはせいぜい2つの最も近い隣接パイプラインとの大域的水平方向連絡を有する。レンダリング・パイプラインは、対応する3Dメモリ・ユニットからボクセル・データを受け取り、ボリューム・データセットの面と整合する二次元(2D)ベースプレーン画像を生成する。ジオメトリI/Oバスが、複数のレンダリング・パイプラインとジオメトリ・エンジン間の大域的水平方向連絡を提供し、ジオメトリI/Oバスが、単一の画像におけるジオメトリック、ボリュメトリックのオブジェクトのレンダリングを可能にする。

#### 【0016】

本発明の装置および方法は、性能、画像レンダリング品質、融通性および単純性の向上のみならず、単一の画像におけるボリューム、表面(特に半透明性)を結合する能力に関しても、既存の3Dボリューム視覚化アーキテクチャおよび方法に勝る。本発明は、任意視野方向からの融通が利き、高品質な真のリアルタイム・レンダリングおよび高解像度データベースの内部、表面構造を視覚化する機構を提供する。そして、さらに、正確な透視投影、マルチ解像度ボリューム、多段オーバーラップ・ボリューム、クリッピング、勾配算出の向上、デプス・キューイング、ヘーズ、スーパーサンプリング、大きなボリュームの異方性データセットおよびレンダリングを含む様々なボリューム・レンダリング強化を支援する。

#### 【0017】

本発明は、単なるボリューム・レンダリング・マシンを超えたもの以上のものであり、高性能補間エンジンであり、たとえば、2D、3Dテクスチャ・マッピング(ミップ・マッピング)および画像ベース・レンダリングを含む補間にかな



り依存している個別の画像形成作業の高解像度ボリューム・レンダリングおよび加速のためのハードウェア・サポートを提供する。さらに、本発明の、ジオメトリ・エンジンと接続された装置および方法は、ボリュメトリック、ジオメトリックの方法を結合し、ユーザが、伝統的なジオメトリック・プリミティブ（たとえば、ポリゴンのファセット）、画像および単一の画像（ユニバーサル3Dレンダリングとして定義される）における画像、ボリュームを一緒に含む複雑なシーンを効果的にモデル化し、レンダリングすることができる。

【0018】

本発明の装置は、それに加えて、種々の大域的、局所的なフィードバック・コネクションを含むことによってシステム融通性を向上させ、これが、画像形成ワーピングおよび多解像度ボリューム処理のような進歩した画像形成作業を実施するようにパイプライン・ステージを再構成する能力を加える。さらに、本発明は、経済的な方法でこれらの目的を達成する。

【0019】

本発明の利点は、ジオメトリ・エンジンに接続したとき、単一画像において一緒にジオメトリおよびボリュームの混合を可能にする方法および装置を提供することにある。

【0020】

本発明の別の利点は、オーバーラップしているボリュメトリック・オブジェクトを効果的にレンダリングする方法および装置を提供することにある。

【0021】

本発明のまた別の利点は、高度なボリューム・レンダリング機能を実施するために選択的なパイプライン・ステージを再構築する能力を有する装置を提供することにある。

【0022】

本発明の更なる利点は、補間にかなり依存する強化像形成作業を支援する方法および装置を提供することにある。

【0023】

本発明のまた更なる利点は、大きいボリューム・データセットの処理を加速す

る方法および装置を提供することにある。

【0024】

本発明の別の利点は、既存の特殊目的ハードウェア・ボリューム視覚化システムよりも高い品質または迅速な透視投影を提供することにある。

【0025】

本発明のさらに別の利点は、勾配評価、補間ユニットに対する強化と共に、画像の品質を容易に向上させることができる方法および装置を提供することにある。

【0026】

本発明のまた別の利点は、ハードウェアで画像ワーピングを実施することのできる装置を提供することにある。

【0027】

本発明のまた更なる利点は、公知のボリューム視覚化システムに固有の欠点を克服する方法および装置を提供することにある。

【0028】

本発明のこれらおよび他の特徴および効果は、添付図面に関連して読むべき実施例の以下の詳細な説明から明らかになる。

【0029】

(好ましい実施例の詳細な説明)

本発明の装置および方法は、データを処理し、高解像度ボクセル・ベース・データセットのリアルタイム視覚化を支援することができる。本発明は、ジオメトリ（たとえば、ポリゴン）で画像（たとえば、ボリューム、テクスチャおよび画像）の統合に加えて強化されたボリューム・レンダリングをデリバリーするユニバーサル三次元（3D）レンダリング・システムである。この装置および方法は、本願でも発明者として名を連ねているDr. Arie Kaufmanの発行済みの特許および係属中の出願、たとえば、米国特許第5,038,302号として1991年8月6日に発行された「Method of Converting Continuous Three-Dimensional Geometrical Representations Into Discrete Three-Dimensional Voxel-Based Representations Within a Three-Dimensional Voxel-Based System」、米国特

許第4, 987, 554号として1991年1月22日に発行された「Method of Converting Continuous Three-Dimensional Geometrical Representations of Polygonal Objects Into Discrete Three-Dimensional Voxel-Based Representations Thereof Within a Three-Dimensional Voxel-Based System」、米国特許第4, 985, 856号として1991年1月15日に発行された「Method and Apparatus for Storing, Accessing, and Processing Voxel-Based Data」、'593出願の継続出願として1993年3月15日に出願された合衆国出願通し番号08/031、599号によって放棄された合衆国出願通し番号07/347、593号として1989年5月4日に出願された「Method of Converting Continuous Three-Dimensional Geometrical Representations of Quadratic Objects Into Discrete Three-Dimensional Voxel-Based Representations Thereof Within a Three-Dimensional Voxel-Based System」、米国特許第5, 101, 475号として1992年3月31日に発行された「Method and Apparatus for Generating Arbitrary Projections of Three-Dimensional Voxel-Based Data」、合衆国出願通し番号08/097、637号として1993年7月26日に出願された「Method and Apparatus for Real-Time Volume Rendering From An Arbitrary Viewing Direction」、合衆国出願通し番号07/85523番として1992年3月20日に出願された「Method and Apparatus For Generating Realistic Images Using a Discrete Representation」および合衆国出願通し番号08/910、575号として1997年8月1日に出願された「Apparatus and Method for Parallel and Perspective Real-Time Volume Visualization」に記載されているようなボクセル・ベース・システムとして使用できるように設計してある。これらの引用文献の全開示内容は、参考資料としてここに援用する。

#### 【0030】

図2は、本発明の一実施例に従って形成したユニバーサル3Dレンダリング・システム10の概念図を示す。レンダリング可能なオブジェクトの集合体を表示するアプリケーション12は、アプリケーション・プログラム・インタフェース(API)14によって適切な画像およびジオメトリ表現に分割されると好まし

い。これらの表現は、それぞれ、引き続いて、画像ユニット16およびジオメトリ・ユニット18によって処理される。これらのユニットは、機能ブロックとして全体的に図示してある。画像ユニット16は、好ましくは、複数の画像パイプライン（図示せず）を包含し、ジオメトリ・ユニット18は、好ましくは、それぞれ、画像およびジオメトリ表現をレンダリングするための複数のジオメトリ・パイプラインを包含する。画像ユニット16およびジオメトリ・ユニット18の描写出力は、ブレンディング・ユニット20において結合され、単一のベースプレーン画像を生成する。このベースプレーン画像は、好ましくは、表示のためにワープ単位22によって最終的な投影プレーンに変換されるとよい。

#### 【0031】

図3は、本発明のCube-5ボリューム視覚化システムの1つの実施例を示す。図3に示すように、このシステムは、好ましくは、1つまたはそれ以上の三次元メモリ・ユニット24を包含し、各3Dメモリ・ユニット24が、入力バス26および対応するCube-5チップ28に垂直方向に接続される。複数のCube-5チップ28が、フレーム・バッファ・ピクセル・バス34に接続した状態で示してある。さらに、本発明のシステム10は、好ましくは、少なくとも1つの普通のジオメトリ・エンジン30とホストコンピュータ32に接続しており、これらは、共に、本発明のCube-5装置と連絡するために入力バス26とフレーム・バッファ・ピクセル・バス34との間に作動可能に接続されている。

#### 【0032】

次に図4を参照して、本発明の装置10は、画像入力バス26に好ましくは接続した複数の3Dメモリ・ユニット24を包含しており、3Dメモリ・ユニット24間の大域水平方向連絡を行うようになっている。ボリューム・データセットは、普通、フル3Dラスタ（すなわち、ボリューム・バッファ）としてしばしば記憶されるボリューム要素すなわちボクセルの正規グリッドとして表される。このボリューム・データセットは、好ましくは、3Dメモリ・ユニット24を横切って分散されると好ましい。スキュード分布の場合、本発明は、主要軸線の内の任意の軸線に対して平行してボクセルの完全なビーム（すなわち、列）への競合

なしアクセスを可能にする。図4に示すように、システム10を通してビデオまたは4次元(4D)ボリューム・データをストリーミングするために、各3Dメモリ・ユニット24は、好ましくは、専用のリアルタイム入力部36に接続される。リアルタイム入力ソースに専用の接続を行うことによって、メモリ・プロセッサ・バンド幅によるボトルネックがさらに軽減される。

#### 【0033】

本発明のユニバーサル3Dレンダリング・システム10は、さらに、複数のレンダリング・パイプラインを包含する。これらは、図4においてCube-5ユニット38の機能ブロックとして示してある。各レンダリング・パイプライン38は、対応する3Dメモリ・ユニット24に接続してあり、好ましくは、少なくとも好ましくはその2つの最も近い隣接3Dメモリ・ユニットと水平方向に連絡する。Cube-5ユニット38は、それらの専用の3Dメモリ・ユニット24から読み出し、二次元(2D)ベースプレーン画像を生成する。このベースプレーン画像(Cube-5ユニット38によって生成した複数の合成ピクセルを含む)は、複数の二次元(2D)メモリ・ユニット40を横切って分散されると好ましい。2Dメモリ・ユニット40の各々は、好ましくは、2Dメモリ・ユニット40間の大域水平方向連絡を提供する対応するCube-5パイプライン・ユニット38およびベースプレーン・ピクセル・バス42に接続している。

#### 【0034】

好ましくは、本発明は、ベースプレーン・ピクセル・バス42に接続された複数のワーブ単位44を包含する。ワーブ単位44は、複数の2Dメモリ・ユニット40に記憶されたベースプレーン画像を組み立て、ユーザ定義画像プレーン上へ変換する(すなわちワーブする)。本発明は単一のワーブ単位44を使用すること(たとえば、ハードウェアのコストまたはオーバーヘッドを低減するために)を意図しているが、複数のワーブ単位44が画像変換を加速すると望ましい。

#### 【0035】

各ワーブ単位44の出力部は、好ましくは、フレーム・バッファ・ピクセル・バス34に接続してあり、ワーブ単位44間の大域水平方向連絡を行う。ベースプレーン・ピクセル・バス42を通してソース・ピクセルを読み込むことと、フ

レーム・バッファ・ピクセル・バス34を通して最終的な画像ピクセルを書き込むことは、同時に生じ、より大きいシステム・スループットを可能にすると好ましい。好ましいアーキテクチャではないが、本発明は、また、ワーブ単位44による逐次的読み書きも意図している。こうすれば、この1つのピクセル・バスが、フル・スクリーン画像のためのリアルタイム画像転送にとって十分なバンド幅を提供すると仮定してのことだが、ピクセル・バスがたった1つでもよい。

#### 【0036】

図4を続けて参照して、本発明は、好ましくは、ジオメトリ入力バス46とジオメトリ出力バス48とを包含するが、2つのバスを、リアルタイム像形成にとって十分なバンド幅の単一のジオメトリ入出力バスにまとめることも考えられる。ジオメトリ入力、出力バス46、48は、それぞれ、Cube-5ユニット38の入力部、出力部に接続してあり、少なくとも1つのジオメトリ・パイプラインまたはエンジン（図示せず）を本システム10に独特な方法で接続すると好ましい。本発明のアーキテクチャ（ジオメトリ・バス46、48を経てジオメトリ・エンジンに接続している）は、画像（たとえば、ボリュームおよびテクスチャ）のジオメトリ（たとえば、ポリゴンおよび表面）との統合をサポートする。ジオメトリ・でえたのボリュームメトリック・オブジェクトとのこの混合は、本発明に特有の強力な特徴である。

#### 【0037】

次に図5を参照して、ここには、本発明の一実施例に従って形成された複数のCube-5レンダリング・パイプライン（図4の参照コード38）のうちの1つの機能ステージを表すブロック図が示してある。図5に示すように、各レンダリング・パイプライン52は、好ましくは、4種類の処理ユニットを包含する。すなわち、トリリニア補間ユニット（TriLin）54、勾配評価ユニット（Gradient）56、シェーディング・ユニット（Shader）58および合成ユニット（Compos）60である。これらレンダリング・パイプライン・ステージの各々は、従来の立方体ボリューム視覚化アーキテクチャ（先に延べた）に関するArie Kaufmanの先に発行された特許および係属中の出願に詳しく記載されており、したがって、ここでは簡単に以下に説明する。

## 【0038】

3Dメモリ・ユニット24に関して上述したように、ボリューム・データセットは、対応する3Dメモリ・ユニット24（図4参照）に接続した各Cube-5ユニット38と共に、3Dメモリ・ユニット24を横切ってスキューした状態で分散したボクセルの正規グリッドとして記憶される。同じスキューしたビームのボクセルは、好ましくは、並列に取り込まれて処理され、すべてのCube-5ユニット38を横切って分散させられる。予め定めたベースプレーンに対して平行な（すなわち、所定の視野方向に対して最も垂直であるボリューム・データセットの面と平行な）ボリューム・データセットの連続したスライスは、好ましくは、スキャンライン・オーダーにおいてトラバースされる。再び図5を参照して、アドレス生成・制御ユニット62が、好ましくは、3Dメモリ・ユニット24内へアクセスするためのアドレスを生成する。さらに、アドレス生成・制御ユニット62は、現処理スライスとして最初の処理スライスを指示し、最終的なスライスが処理されてしまうまでボリューム・データセットの引き続くスライスを通しての走査を制御する。

## 【0039】

トリリニア補間ユニット54は、2つの処理スライス間の補間サンプル値の新しいスライスを計算する。あるいは、トリリニア補間機能を一連のリニアまたはバイリニア補間として実施することも考えられる。

## 【0040】

勾配評価ユニット56は、好ましくは、ボリューム・データセットの複数のスライスからのボリューム・データを使用して中央差分勾配を計算する。勾配評価ユニット56によって生成した中央差分勾配を利用して、引き続き、現処理スライスのサンプル・ポイントにシェーディング・ユニット58が陰影を付ける。シェーディング・ユニット58は、好ましくは、1つまたはそれ以上のルックアップ表（LUT）（好ましくは、材料カラーおよび強度情報を記憶している各シェーディング・ユニット58に存在するもの）へのインデックスとしてサンプルおよび勾配を使用する。材料カラー表は、データセット・タイプ依存であり、カラー強度表は、当業者には公知のように、局所的照明モデルに基づいている。簡単に

言えば、カラーおよび強度の乗算が、各サンプルについてのピクセル・カラーを生成し、これが、合成ユニット60において使用され、このカラーを各サイト・レイに沿った先に蓄積されていたピクセルと合成する。

#### 【0041】

再び図4を参照して、連続するサイト・レイに沿った次のサンプルを計算するためのデータは、隣接したCube-5ユニット38上に存在する可能性がある。この場合、Cube-5ユニット38間の最も近い隣接のコネクションが、好ましくは、必要なデータを適切なCube-5ユニット38に送るのに用いられ、このCube-5ユニット38が、この特定のサイト・レイを処理し続けることになる。合成が完了したとき、合成されたピクセル（すなわち、ベースプレーン・ピクセル）は、好ましくは、Cube-5ユニット・パイプライン38に接続した対応する2Dメモリ・ユニット40に記憶される。ベースプレーン・ピクセル（ベースプレーン画像を形成している）が、その後、ベースプレーン・ピクセル・バス42を経て2Dメモリ・ユニット40から読み込まれ、ワープ単位44によって組み立てられる。ワープ単位44は、それに加えて、最終的な投影プレーン画像にベースプレーン画像を変換する。

#### 【0042】

図5を参照して、トリリニア補間ユニット54および勾配評価ユニット56にとって必要なデータ遅延は、好ましくは、トリリニア補間54および勾配評価ユニット56によって処理される前に1つまたはそれ以上の先入れ先出し（FIFO）ユニット64をパイプライン・データ経路に挿入することによって達成される。FIFOユニット（単数または複数）64は、たとえば、ランダム・アクセス・メモリRAM（好ましくは、Cube-5チップに埋め込まれている）として実装してもよい。ボリューム・データセットの複数のスライスのビームを同時に処理するとき、したがって、スライス間により長い計算時間が必要なときには、所定遅延の導入は特に重要である。

#### 【0043】

バイリニア補間ユニット（BiLin）72に作動可能に接続した合成バッファ（Compos Buffer）74が、本質的に、1つのスライスFIFOを提供する。バイリ



ニア補間ユニット72は、好ましくは、テクスチャ・マッピングのために必要に応じてボクセル間の値を得るように補間する。ボリューム・レンダリングのために、BiLin72は、好ましくは、ボリューム・データセットのコーナー・ボクセルの1つを選ぶ(Select xおよびSelect yによって決定される)0.0または1.0の重みだけを使用する。レイがパイプラインと交差する場合、レイ・データだけを移動させる。xおよびyについての $\mu x[?]$ だけで、ボリューム・レンダリングについては充分であるが、テクスチャ・マッピングにとってはバイリニア補間が好ましい。

#### 【0044】

Cube-5アーキテクチャは、好ましくは、パイプライン・ステージの再順序付けおよびマルチパス・レンダリング、処理動作の数をサポートする。この場合、Cube-5レンダリング・パイプライン52および3Dメモリ・ユニット24の種々のステージ間にフィードバック・コネクションが必要となる。たとえば、オーバーラップしているボリュメトリック・オブジェクトの正しいレンダリングには、好ましくは、Cube-5パイプライン52を通る少なくとも2回のパスが必要であり、このとき、一回目のパスでボリュメトリック・オブジェクトを再サンプリングして互いに整合させ、二回目のパスでインタリーブ・オーダー式にボリュメトリック・オブジェクトをレンダリングする。図5に示すように、多ボリューム・フィードバック経路66を設けると好ましく、これが合成ユニット60の出力部を対応する3Dメモリ・ユニット24に作動可能に接続する。これにより、再サンプリング・ボリュームを、再サンプリング、分類、シェーディング後に3Dメモリ・ユニット24に書き戻すことができる。最終的なレンダリング・パスは、RGB $\alpha$ ボリュームについて作動する。

#### 【0045】

同様に、各Cube-5レンダリング・パイプライン52は、好ましくは、ワーブ単位44と3Dメモリ・ユニット24の間に接続した画像ベースのレンダリング・フィードバック経路68を包含する。画像ベースのレンダリング・フィードバック・ライン68は、好ましくは、中間ワーブド画像を3Dメモリ・ユニット24に書き込むためのフィードバック経路を提供する。これは、特に、複数の

ワープ・パスを必要とする或る種の画像ベース・レンダリング作業を加速するために有用であるかも知れない。本発明のアーキテクチャは、さらに、3Dメモリ・ユニット24と他の種々のCube-5レンダリング・パイプライン・ステージとの間、または、個々のパイプライン・ステージそれ自体の間のフィードバック・コネクションも意図している。画像レンダリング速度は、フィードバック経路を含むことによってかなり速くなり、これは、Cube-5レンダリング・パイプライン52全体を通して移動するように結果を待つことなく、個々のパイプライン・ステージの計算結果への直接的で即時のアクセスを提供する。

#### 【0046】

本発明の好ましい実施例においては、Cube-5システムは、レンダリング・パイプラインの選択的なステージを迂回するコネクションを包含する。これは、たとえば、或る種の像形成作業にとっては不要であるかも知れない。これらの未使用のパイプライン・ステージを迂回することによって、このような像形成作業が加速される可能性はある。図5に示すように、テクスチャ・マップ・バイパス70は、好ましくは、各Cube-5レンダリング・パイプライン52に含まれる。このテクスチャ・マップ・バイパス・コネクション70は、実質的にミップ・マッピングの速度を上げる。このミップ・マッピングは、たとえば、シェーディング・ユニット58および合成ユニット60を迂回し、トリリニア補間ユニット54および勾配評価ユニット56からの結果をバイリニア補間ユニット72に直接与えることによって処理されるべき画像の複数のディテール・レベル（LOD）を記憶することからなる。このような方法において、本発明のアーキテクチャは、好ましくは、ボリューム・レンダリングを実施するためのパイプライン・アレイとしてばかりでなく、様々な像形成作業を実施するように選択的に構成し得るハードウェア・リソースの集まりとしても考えることができる。たとえば、本発明のCube-5システムがボリューム・レンダリングを実施しているとき、本質的に、ハードウェア・リソースのすべてが必要であるのに対し、テクスチャ・マッピングは、一般的に、メモリ、或る種のバッファリングおよび補間ユニットだけしか必要としない。

#### 【0047】

以下に説明する本発明の別のユニークで重要な局面は、Cube-5アーキテクチャが少なくとも1つの普通のジオメトリ・エンジン76とインタフェース接続していて、単一画像におけるジオメトリック・データとポリュメトリック・オブジェクトとの混合をサポートすることができるということにある。これは、好ましくは、上述したように、少なくとも1つのジオメトリ・バスを設けてジオメトリ・エンジン76とインタフェース接続させることによって達成される。

#### 【0048】

好ましくは、本発明のCube-5アーキテクチャは、可能なときにはいつでも、パイプライン構成要素（たとえば、補間ユニット等）を再利用し、複数の構成、特に、複数のポリュメトリック・オブジェクト、ポリゴン・オブジェクトのシーンのレンダリング、テクスチャ・マッピングおよび画像ベース・レンダリングを使用して様々なレンダリング・アルゴリズムを加速するようになっている。他の重要な利点としては、パイプライン構成要素を再利用することで、ハードウェア・コストを低減できるということがある。Cube-5アーキテクチャは、また、ポリューム・レンダリングを向上させ、他の像形成作業を加速させるための種々のユニークな方法およびアルゴリズムをサポートする。これらの方法およびアルゴリズムのうちのいくつかを、以下に個別にさらに詳しく説明する。

#### 【0049】

本発明に従って形成した、Cube-5システムの好ましい実施例においては、ポリューム・データセットはブロックとして記憶され、それによって、空間的場所についての利点を得ている。リニア・ブロッキング（たとえば、ボクセレータAPI）の代わりに、階層的なブロックを使用している。これは、好ましくは、複数の3Dメモリ・ユニットを横切って分散された配置で記憶され、スキューさせられる。たとえば、現三菱電機16ビット、125メガヘルツ同期ダイナミック・ランダム・アクセス・メモリ（SDRAM）を使用して3Dメモリを実装した場合、各ブロックは、1024バイトを必要とする $8^3$ 16ビット・ボクセルまたは2つのSDRAMページを収容し得る。

#### 【0050】

各ブロックは、好ましくは、同じ3Dメモリ・ユニットに存在している $2^3$ ボ

クセル・ミニブロックとして構成される。SDRAM内部のバンクは、好ましくは、パイプライン式にアクセスされ、現バースト転送がほぼ完全に以降のバースト転写のセットアップを隠すようにすることができる。ミニブロック内のボクセルのビュー依存処理オーダーがそれらの記憶オーダーと一致しない場合には、8つのミニブロック・ボクセルをCube-5チップで再オーダーすると好ましい。それ故、SDRAM上のボリューム・データセットのコピーは、たった1つで充分である。したがって、階層的なブロッキングにより、ほぼフル・バースト・モード速度、ほぼフル(100%)バンド幅利用、ビュー独立データ記憶およびバランスの取れた作業負荷で、ミニブロックにランダム・アクセスすることが可能になる。

#### 【0051】

ブロッキングは、メモリ・インタフェースを最適化するだけではなく、チップ間連絡バンド幅(すなわち、Cube-5ハードウェア・ユニット間の)を縮小するという付加的な利点も有する。これは、ブロック周辺部にあるボクセルだけが隣接したチップス処理隣接ブロックとの間で交換される必要があるからである。 $O(b^3)$ 時に $b^3$ ボクセル・ブロックを処理しながら、ブロック境界上の $O(b^2)$ ボクセルはチップス処理隣接ブロック間で連絡する必要があるだけである。ここで、 $b$ はブロック・エッジのサイズであり、各ブロックは、 $b \times b \times b$ (すなわち、 $b^3$ )のボクセルを有する。したがって、チップ間連絡は、ノンブロッキング解の場合よりも $O(1/b)$ 小さいバンド幅でよい。ブロック・エッジのサイズ $b$ は、約 $4 \leq b \leq 64$ の範囲にあってもよいが、8のブロック・エッジ・サイズが好ましい。

#### 【0052】

ブロック・ルックアップ表(LUT)は、現ボリュームを含むすべてのブロックへポインタを記憶するのに利用されると好ましい。この方法は、大きいボリュームの当該選択領域にズーミングしながら能動ボリュームを制限する簡単な方法を提供する。また、任意に形付けしたサブボリューム(ブロック・サイズの細分性で)のレンダリングも可能にする。その上、多くの小さいボリュームを収容しているシーンは、非常に効率よくレンダリングされ得る。それは、すべてのボリ

ュームが3Dメモリ・ユニットの中のどこにでも存在できるし、3Dメモリ・ユニットよりもむしろ、ルックアップ表だけを各ボリウムについて再ロードすれば良いからである。

#### 【0053】

透視投影および／またはディテール・レベル（LOD）を実施する1つ方法は、x、y方向における2倍スーパーサンプリングに依存する。したがって、トリニア補間用の補間ユニットならびに勾配計算用の勾配評価ユニットの4倍の複製を使用すると好ましい。その結果、SDRAM、Cube-5パイプライン間のデータパスはほぼ不変である。しかしながら、Cube-5パイプライン間のバンド幅は、オンチップ・スループットおよびバッファと同様に、4倍になる。その主たる理由は、ノーマル・モードの各サンプルが最高4つのサンプルと交換されるからである（すなわち、x方向において倍速、y方向において倍速）。

#### 【0054】

好ましくは、異方性データセットの取り扱いおよびスーパーサンプリングには、不透明度 $\alpha$ の修正を行うとよい。結合された機能は、 $\alpha' = 1 - \alpha)^{d/k}$ 、ここで、スーパーサンプリング・ファクタ $k$ は、1ボクセル・セル当たりのサンプルの数を表しており、 $d$ は、各ボクセル・セルを通してサイト・レイが移動する距離（すなわち、サイト・レイの経路長）を表している。好ましくは、レンダリング中に、 $\alpha'$ の急速ルックアップのために、ルックアップ表（LUT）を使用する。

#### 【0055】

図5を続けて参照して、下層のボリウム・データセットの均一なサンプリングで終わるボリュメトリック・データの透視レンダリングでは、レベル間でフィルタリングを行いながら合成バッファ74の再スケーリングが必要である。ディテール・レベル（LOD）透視レンダリングでは、レベル間での合成バッファ74の再アラインメントが必要である。これらのプロセス（パイプライン52で利用できない大域的連絡を取り入れている）は、共に、好ましくは、ワープ単位（単数または複数）44によって実施される。合成バッファ74は既にワープ単位44にアクセス可能となっているが、フィードバック・ライン43を用いて合成

バッファ74へ濾過された値を書き戻すと好ましい。

#### 【0056】

リアルタイム（すなわち、30ヘルツのフレーム率）で最終的なフル・スクリーン画像を得るのにハードウェア・ワープ単位が必要である。図5に示すように、ベースプレーン画像（Cube-5レンダリング・パイプライン52の合成ユニット60によって生成した画像）は、2Dメモリ・ユニット40においてバッファされると好ましい。2Dメモリ・ユニット40からワープ単位44までのメモリ・バンド幅を下げるために、ベースプレーン画像の各ピクセルは、一度だけアクセスされると好ましい。現および前のスキャンラインのサンプル間のリニア補間を実施するために、前のスキャンライン・サンプルを記憶するには、少なくとも1つのスキャンラインを保持するようなサイズとした別のFIFOユニットが必要である。各グリッド・ピクセルについての補間重みは、ホスト・マシンで事前計算すると好ましい。

#### 【0057】

不透明なジオメトリック・オブジェクトについてボリュームおよびジオメトリを正確に混ぜるために、Zバッファ画像は、合成バッファ60に書き込まれると好ましい。合成ユニット60は、各新しいサンプルを混合する前にz比較を実施しなければならない。その上、半透明のジオメトリック・スライスについては、ジオメトリ・エンジン76は、好ましくは、本発明のジオメトリ入力バス（図4の参照コード46）を利用して、RGB $\alpha$ 値の各スラブをデータ流に挿入し、その結果、各スラブがボリュメトリック・データ・スライスでインタリーブされる。

#### 【0058】

テクスチャ・マッピングについて、たとえば、図6が、本発明に従って、32ビットのテクセル・データを3Dメモリ・ユニットの16ビット・ボクセルのミニブロックにおける2×2近辺にどのように記憶させられるかを示している。したがって、32ビット・テクセルの4テクセル近辺が、各メモリ・バースト読み込み中に読み込まれると好ましい。データ重複がない場合、いくつかのテクスチャ座標が記憶済みのミニブロック間に位置し得るので、平均2.25データ・バ

ースト読み取りで、適切なテクセル近辺にアクセスするようにCube-5システムが実施すると好ましい。

#### 【0059】

再び図5を参照して、本発明の1つの形態に従って、ハードウェアで画像ベースのレンダリングを実施する1つ方法では、メモリ制御機構78を利用する。この機構は、各Cube-5パイプライン52に内蔵され、各パイプラインについての寄与領域に基づいて適切なソース・ピクセルを読み込むと好ましい。このパイプライン52における補間ユニット（たとえば、54、72）が、次に、明視野レンダリングまたはルミグラフィにとって必要とされる4次元（4D）補間を実施すると好ましい。別の実施例として、ワーブ単位44を利用してこの機能を実施してもよい。現視野に寄与しているソース・ピクセルが、好ましくはコネクション・ライン41を通して2Dメモリ・ユニット40に読み込まれ、組み立てられ、その後、ワーブ変換が行われる。好ましくは、4つの組み立てられたソース画像は、4つの連続したワーブ・パスにおいて処理される。4つの中間ワーブド画像の最終的な組み合わせは、Cube-5パイプライン52で実施される。先に説明したように、画像ベースのレンダリング・フィードバック・ライン68は、中間ワーブド画像を3Dメモリ24に書き込むためのフィードバックを提供する。いずれの方法についても、3Dメモリ・ユニット24は、大きい画像データベースのための局所的記憶装置を提供する。

#### 【0060】

ここで、上記の（Cube-5と呼ぶ）本発明の装置は、既に説明したユニバーサル・レンダリング以上に、普通のボリューム処理方法をかなり加速することができることは了解されたい。その上、本発明の装置は、性能を強化するために構成される多くのユニークなアルゴリズムと連動して使われてもよいCube-5および／またはリアルタイム・ボリューム処理の性能を強化するおよび／またはリアルタイム・ボリューム処理のための強化した特徴を提供するようになっており、したがって、Cube-5システム全体を既存のボリューム・レンダリング・アーキテクチャ、たとえば、Cube-4より優れたものとすることができる多数のユニークなアルゴリズムと関連して使用することができる。画像ワーピ

ング、三次元変換、透視投影、大ボリューム処理、高品質レンダリング、クリッピング、デプス・キューイング、スーパーサンプリングおよび異方性データセットを実施するためのものを含むこれらユニークなアルゴリズムのいくつかを以下に詳しく説明する。

#### 【0061】

本発明の1つの形態によれば、画像ワーピングを実施する方法が提供される。この方法は、他の利点のうちでも、透視ワーピングの速度を上げ、向上する画質を提供する。画像ワーピングは、好ましくは、Cube-5ボリューム・レンダリング・パイプラインの最終ステージである。簡単に言えば、画像ワーピングは、主として、2つの画像、すなわち、ソース画像と目標画像の間のジオメトリック変換に関する。ジオメトリック変換は、ソース・ピクセルと目標ピクセルとの関係を定める。効率および高品質は、このような用途で等しく重要な問題である。本発明の装置においては、ワープ単位は、好ましくは、画像変換機能を実施する。その結果、ワープ単位を使用する用途は、本発明の画像ワーピング方法から利益を得る。

#### 【0062】

変換のデータフローによって区別すれば、画像ワーピング方法は、一般に、順方向ワーピングあるいは逆方向ワーピングのいずれかとして分類される。順方向ワーピングにおいては、ソース・ピクセルがスキャンライン・オーダーで処理され、結果が目標画像上へ投影される。逆方向ワーピングにおいては、ラスタ・オーダーにおける目標ピクセルが、逆にソース画像にマッピングされ、したがって、サンプリングされる。大部分の公知従来技術ワーピング・アルゴリズムは、逆方向ワーピングを使用する。

#### 【0063】

アフィン変換（すなわち、並進、回転、スケーリング、シャーリング等）と比較した場合、透視変換は、より高価で挑戦するに値すると考えられる。透視投影の場合、投影プレーンにおける1つのピクセルについてベースプレーン画像におけるサンプル場所を計算するときに、高価な除算が必要である。CPUによって実施するときには、普通の透視ワーピングは、代表的には、平行ワーピングより



も少なくとも三倍遅くなる。したがって、いくつかの従来技術方法は、透視変換を複数のパスを必要とするいくつかの単純な変換に分解していた。しかしながら、マルチパス変換アルゴリズムに固有の1つの主要な問題は、2つ一次元の(1D)フィルタリング作業の組み合わせが、真の二次元(2D)フィルタリングほぼ融通性がないということである。さらに、普通のマルチパス方法は、画質を劣化させる付加的なフィルタリング作業を行う。

#### 【0064】

本発明は、好ましくは、実質的にアフィン変換と同じ効率で実施することができるユニークな単パス順方向ワーピング方法を使用する。すべてのピクセルに対して伝統的に実施されてきた高価な除算は、本発明によれば、1スキャンライン当たりたったの二回に減らされる。したがって、除算作業の数を減らすことによって、本発明は、たとえば、少なくとも速度および効果的なハードウェア実施に関して従来公知の技術による方法よりも優れた、代わりとなり得る透視ワーピング方法を提供する。本発明による、透視ワーピングのための好ましい方法を以下に説明する。

#### 【0065】

好ましくは、本発明は、透視ワーピングを実施するのにスキャンライン方法を使用する。しかしながら、ノーマル・ラスタ・スキャンライン・オーダーにおいて走査するよりもむしろ、本発明のアルゴリズムは、ソース画像における特殊なスキャンライン方向において処理される。図7、8に示すように、この特殊なスキャンライン方向92(図8)は、好ましくは、ソース画像80の平行スキャンライン84が、目標画像82における平行スキャンライン86として現れるという特性を有する。そして、ソース・スキャンライン84に沿った等距離サンプル・ポイント88が、目標スキャンライン86においても等距離サンプル・ポイント90として残るという特性を有する。このユニークな方法のいくつかの利点としては、透視・補正画像ワーピングの複雑さの低減(すなわち、1ピクセルあたりの除算をなくし、1スキャンラインあたり2回の除算に代えることによる)、異方性フィルタリングを取り入れることによる正確なアンチエイリアシング、バイリニア補間によって生じたグーロー・シェーディングにおける欠点の訂正およ

び各ソース・ピクセルを正確に一度だけ読み込むことによるメモリ・バンド幅の最適化がある。

#### 【0066】

図8に示すように、この特殊なスキャンライン方向の直観は、投影ジオメトリに由来する。図8を参照して、ソース画像80は、好ましくは、三次元(3D)表面に置かれ、そして、目標画像82は、スクリーン上に置かれる。代表的なテクスチャ・マッピングと同様に、スクリーン上にピクセルを得るために、サイト・レイ(単数または複数)94が、視点(またはアイ・ポイント)から3Dスペースにキャストされ、スクリーン82および3D表面80と交差させられる。交差点が、サンプル・ポイント98である。スクリーン・スペースにおける走査方向92が3D平面と平行であるとき、両方の画像のスキャンラインは互いに平行である。そして、スキャンラインに沿った等距離のサンプル・ポイント98は3D表面プレーンにおいても等距離のままである。この平行・保存(PP)スキャンライン方向が、所与の透視変換のために存在し、ユニークである。平行投影の場合、任意の走査方向が、両方の画像上にこの平行性を保存し、その簡略性によりラスタ・スキャンライン方向を使用するのが好ましいかも知れないことは了解されたい。

#### 【0067】

再び図7を参照して、平行・保存(PP)スキャンライン84、86は、それぞれ、ソース画像80、目標画像82の両方において示してある。ひとたび平行性を正しく達成したならば、ピクセル・アクセスは正規となり、そして、空間干渉性を両方の画像において利用することができる。その上、PPスキャンラインは、ソース・サンプル88の投影を算出するために各スキャンラインに対して除算を行うことなく、純粋増分アルゴリズムの適用を可能にする。しかしながら、非リニア投影によるすべてのスキャンラインの2つのエンドポイントのために、なお一回の除算は必要である。

#### 【0068】

図7を続けて参照して、ソース画像80を、ラスタ方向よりもむしろPPスキャンライン方向において走査したとき、目標スキャンライン86上のサンプル・

ポイント90は、必ずしも目標ピクセル91と一致する必要はない。しかしながら、サンプル・ポイント90は、目標画像82のxグリッドライン89上で整合させることができ、したがって、サンプル・ポイント90は、yグリッド・ライン87からのみ外れている（スキャンラインに沿って等距離である）。より効果的ではあるが、品質が低い実施例の場合、半ピクセルが最大エラーであるから、最も近い隣接の目標ピクセルにサンプル値を置くことが合理的近似である。しかしながら、より高い品質が好ましい場合には、本発明は、以下に説明するピクセル訂正および効果的アンチエイリアシングを実施することができる。

#### 【0069】

一般に、 $0(n^2)$  から  $0(n)$  への除算の回数の低減は、本発明のアルゴリズムによって得られる（ここで、 $n$  は、リニア解像度である）。本アルゴリズムの場合、好ましくは、各サンプル・ポイントを算出するためには2回だけの加算が必要であるが、普通のラスタ・スキャンライン・アルゴリズムでは、一般に、1ピクセル当たり3回の加算、1回の除算および2回の乗算が必要である。本発明による、順方向画像ワーピングを実施する好ましい方法を以下に詳しく説明する。

#### 【0070】

本発明の順方向ワーピング・アルゴリズムは、好ましくは、2つのステージにおいて実施される。すなわち、（1）特殊な平行・保存（PP）スキャンライン方向を算出すること、そして、（2）各スキャンライン内で増分状態で、特殊なPPスキャンラインに沿って目標画像にソース画像を順方向マッピングすることである。

#### 【0071】

簡単に上述したように、平行・保存（PP）スキャンラインは、三次元（3D）平面とスクリーン（すなわち、目標画像）との間の交差ラインである。しかしながら、二次元（2D）問題においては、PPスキャンラインは、2Dマトリックスに基づいて算出しなければならない。一般に、透視変換は、次のように表すことができる。

#### 【0072】

#### 【数1】

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = M \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a & d & g \\ b & e & h \\ c & f & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

ここで、(u、v) は、ソース・ピクセルの座標であり、(x、y) は、目標ピクセルの座標であり、Mは、透視変換マトリックスでる。(u、v) 座標は (x、y) によって次のように表現することができる。

【0073】

【数2】

$$(u,v) = F(x,y) = C \begin{bmatrix} ax + dy + g \\ bx + ey + h \end{bmatrix}$$

ここで、

【0074】

【数3】

$$C = \frac{1}{(cx + fy + 1)}$$

目標画像におけるラインは、 $y=kx+B$  として表現することができる。ここで、勾配kはライン方向を示し、Bはライン切片を示している。PPスキャンラインについての勾配kを算出するためには、好ましくは、0、1の同じ勾配kおよび切片Bを有する2つの平行ラインを定義する。これらは、それぞれ、(0、0)、(1、k)、(0、1)、(1、k+1)のポイント・ペアによって表される。ソース画像におけるこれらのポイントの座標を次に算出する。透視変換が直線を保存するので、これら2つのラインは、ソース画像における直線として残り、それらの勾配は、2つのポイント・ペアから算出することができる。ここで、

2のマッピングされたラインの勾配がほぼ同等であると仮定するならば、kにおける式が好ましくは得られる。kについてのこの式を解いた結果、

【0075】

【数4】

$$k = -\frac{c}{f}$$

となる。ソース画像における対応する勾配k'は、次の通りである。

【0076】

【数5】

$$k' = \frac{bf - ec}{af - dc}$$

上記の式からわかるように、k' = -c/fのとき、同種座標の分母は、Bf + 1の一定値となる。ここで、Bは、y = kx + Bにおける切片である。

【0077】

本発明の好ましい順方向ワーピング方法の第2ステージでは、スキャンライン処理を行う。これは、図9、10に例示してある。次に図9を参照して、好ましいアルゴリズムは、ソース画像80を通してスキャンライン84（たとえば、スキャンラインS1～S4）を走査する。上記のように、スキャンライン84は勾配k'を有する。各スキャンライン84に沿ったサンプル88は、好ましくは、増分算出される。各スキャンライン84について、まず、目標画像からソース画像上へエンドポイントの投影を算出する。次いで、スキャンライン上のサンプル・ポイントの数に基づいて、x、y両方向における増分を算出する。

【0078】

ソース画像におけるサンプルの伝統的なバイリニア補間を考慮した場合、あらゆるサンプルは、本質的に、4つの周囲ソース・ピクセルの寄与を必要とする。ピクセルがすべてのサンプルについて毎回読み込まれる場合、各ピクセルは4回

読み込まれるべきである。これは、目標画像サイズの4倍であるメモリ・バンド幅に通じる。しかしながら、すべてのスキャンラインが平行であるから、隣接したスキャンライン上のサンプルは、通常、寄与しているソース・ピクセルを共有する。したがって、本発明の方法によれば、先に読み込まれていたピクセルは、好ましくは、共通ピクセルがソース画像自体からよりもむしろバッファから読み込まれるようにバッファされる。

#### 【0079】

図7を参照して、ピクセルは、好ましくは、ピクセル読み込みテンプレート100と呼ばれる固定パターンにおいて読み込まれ、Bresenhamライン・アルゴリズム（当業者によって公知である）に基づいて算出される。図7の底に示している二進数は、読み込みテンプレート100をコード化する1つ方法を表している。しかしながら、当業者によって明らかなように、本発明は、他のコード化方式も意図している。図7に示すように、このコードは、正のv方向において増加を示し、「0」は増加を表さず、「1」は1単位ずつの増加を示している。一方、uは、常に1単位ずつの増加を示している。図7の例の場合、u軸線は、好ましくは、主要処理軸線と呼ぶとよいかも知れない。テンプレート100が常に最も左のピクセルから出発し、垂直方向（すなわち、増加するv方向）に移動し、すべてのピクセルが、サンプリングにおける引き続く使用のためにバッファに読み込まれ、そこに置かれると好ましい。2つの点線間の任意のスキャンライン上でのサンプリングのためのピクセルを得るために、たとえ1つの特定のスキャンラインについても4つのピクセル・テンプレートが必要であるが、3つのピクセル・テンプレートだけでも充分であるかも知れない（たとえば、現スキャンラインS2を処理するのにテンプレート2、3、4のみが必要であるということ）ということは図7からわかるであろう。したがって、バッファ・サイズは、好ましくは、4本のスキャンラインである。

#### 【0080】

次に図10Aを参照して、ここには、バッファにおけるサンプルのアドレス指定が示してある。テンプレート・コード値が1であるときはいつでも、サンプルはv方向において1単位ずつ減少する。太いジグザクのライン104は、バッファ

アにおける出力スキャンラインを表している。サンプルが陰影領域106内に入ると（この場合、バッファ内のピクセルがシャーされる）、サンプリングについての正しいピクセルを読み込むことに注意しなければならない。図10Bは、この領域におけるサンプルsの1つをバイリニア的に補間するための好ましい手順を示している。

#### 【0081】

バッファの内容は、好ましくは、スキャンライン位置に基づいて更新される。たとえば、図9を参照して、スキャンラインS1を処理するとき、テンプレート1、2、3、4がバッファ内にあることが好ましい。スキャンラインS2については、バッファは、好ましくは、同じままである。スキャンラインS3については、テンプレート5は、好ましくは、バッファ内に読み込まれ、テンプレート1は廃棄される。スキャンラインS4については、テンプレート6は、好ましくは、テンプレート2を交替する。これ以降は同じである。

#### 【0082】

上述したように、本発明のユニークな順方向画像ワーピング方法における特徴の1つは、グーロー・シェーディングにおける欠点の訂正である。グーロー・シェーディングは、ジオメトリック・オブジェクトの表面に陰影を付けるために用いられるポピュラーな強度補間アルゴリズムである。頂点のところのみカラーを与えた場合、グーロー・シェーディングは、ラスタ・スキャンライン・オーダーでジオメトリの全ラスタライゼーションのために強度をバイリニア的に補間する。グーロー・シェーディング方法における欠陥は、この技術分野において周知であり、たとえば、Digital Image Warping, by G. Wolberg, IEEE Computer Society Press, 1990のような論文の主題となっている。

#### 【0083】

グーロー方法と関連した問題の1つは、斜めのライン（一例として）が透視投影のためにリニアにマッピングされないということである。斜めのラインが3Dスクリーン・スペース上へ透視投影されたとき、グーロー・シェーディングは、この斜めのラインを曲線に変換し、この曲線が、透視変換におけるラインを保存する特性を損なうということである。

## 【0084】

本発明の画像ワーピング方法は、グーロー・シェーディングにおける透視ゆがみを補正するのである。スクリーン・スペースにおけるラスタに沿ったリニア補間が、ジオメトリック座標に変換されたときに、一般的に非リニアとなるために、透視ゆがみが存在する。しかしながら、本発明の特殊な走査方向を使用した場合、直線性はマッピングすることによって保存される。したがって、画像スペースおよびジオメトリック・スペースの両方において補間がリニアとなり、それによって、グーロー・シェーディングのゆがみを固定する。ここで、エッジに沿った補間がなお非リニアであり、したがって、スキャンライン・エンドポイントが正しい補間についてジオメトリック・スペースに変換されなければならないということは明らかであろう。

## 【0085】

本発明の順方向マッピング・アルゴリズムは、最も近い隣接の近似をもって、伝統的な方法を使用して生成した画像からほぼ見分けがつかない目標画像を生成するという点で好ましい。しかしながら、より高い画質が望まれる場合には、本発明の方法は、好ましくは、正確なグリッド・ポイントでピクセル値を算出することができる。単純な目標ピクセル訂正方式は、好ましくは、この訂正を実施するように導入するとよい。

## 【0086】

次に図11を参照して、目標画像82におけるサンプル・ポイント90が整数のx座標上で整合していると仮定した場合、正確なピクセル・グリッド場所91でピクセル値を得るためには、各ピクセルの直ぐ上、下の2つのサンプルのリニア補間が実施されると好ましい。このリニア補間を単に二回目のパスとして実施すると、サンプルを再度読み込まなければならないので、コストが増大する可能性がある。その代わりに、各サンプルが生成される毎に、本発明の好ましい方法は、中間バッファリングをまったく行うことなく、各サンプルの、対応する上下のピクセルへの寄与率を拡張する。

## 【0087】

図11の例で示すように、より太い傾斜したスキャンライン108上に位置す



るサンプル112は、それらに隣接する陰影付きのピクセルに寄与する（スキャンライン上方ではより明るいシェーディングを、スキャンライン下方ではより暗いシェーディングを行う）。矢印は、各サンプル112が、好ましくは、2つのピクセルに寄与することを示している。両方の寄与率が集められるまで、ピクセルが書き出されないことが好ましい。したがって、1つのスキャンライン・バッファは、好ましくは、中間ピクセル値を記憶するために内蔵される。

#### 【0088】

正しくかつ能率的にピクセルを書き出すために、ピクセル書き込みパターン（ピクセル書き込みテンプレート110と呼ばれる）は、好ましくは、事前計算される。ピクセル読み込みテンプレート（たとえば、図9の参照コード100）と異なり、ピクセル書き込みテンプレート110は、好ましくは、1つのスキャンラインに沿ってサンプル値の $y$ 座標値を切り捨てることによって、算出される。テンプレート110は、一連の整数 $y$ ステップおよび真のスキャンライン86からの部分的な距離 $dy$ としてコード化されると好ましい。最終的なリニア補間のために使用される重みは、それぞれ、上下のピクセルについて $dy$ および $1-dy$ である。すべてのスキャンラインが垂直方向（すなわち $y$ 方向）において1単位離れていると好ましいので、テンプレートは1投影当たりたった一度だけ算出される。

#### 【0089】

本発明の順方向ワーピング方法は、アンチエイリアシングによって画質をさらに向上させることができる。平行・保存（PP）スキャンラインを使用することで、より高い品質でより安価なアンチエイリアシング方法を達成することができる。

#### 【0090】

再び図7を参照して、ソース画像の上方スキャンライン上にあるサンプル・ポイントは、下方スキャンライン上にあるよりもまばらであり、アンダーサンプリングからノーマルサンプリングへの移行部を生じている。したがって、適切な再サンプリング・フィルタを使用して上方スキャンラインについてのエイリアシングを避けることができる。等方性フィルタリングは、明らかに、不正確でぼんや

りした画像を生じさせる。異方性フィルタの必要性は、Survey of Texture Mapping, by P. S. Heckbert, IEEE Computer Graphics and Applications, 6 (11) : 56-67, November 1986のような論文、もっと近いところでは、Texram: Smart Memory for Texturing, by A. Schilling, et al., IEEE Computer Graphics and Applications, 16 (3) :32-41, May 1996に示されている。

#### 【0091】

各フィルタがそのフットプリントおよびプロファイルによって定められることは、当業者にとって公知である。目標サンプルを1つの円として仮定すると、ソース画像におけるその投影画像はそのフットプリントである。図12に示すように、このフットプリント114は、一般に、円形（すなわち、等方性）であってはならないし、正方形（すなわち、ミップ・マッピングと同様の形）であってはならず、円錐形でなければならない。フィルタのプロファイルは、フットプリント内の寄与しているピクセルの重みを決定する。sincフィルタが最適であるが、ガウス・フィルタは実施するのがより容易であり、その有限のフットプリントおよび良好なローパス特性のために好ましい。本発明の透視ワーピング・アルゴリズムは、異方性フットプリントを算出する際により高い精度を与え、より低いコストでより高い画質を生じさせる。

#### 【0092】

異方性フットプリントを算出する普通の方法を使用する場合、楕円の主軸を、すべてのピクセルについて算出しなければならない。近似値が提案されたけれども、これは高価な計算を残し、公知の増分方法を利用できないのである。これらの従来技術方法を使用して楕円の主軸を得るためには、ヤコビ行列式を計算しなければならない。しかしながら、本発明の画像ワーピング方法を使用すると、ヤコビ行列式の計算は除いてもよい。

#### 【0093】

本発明に従って異方性フットプリントを計算する好ましい方法に対する洞察を獲得するために、ヤコビ行列式の特性を、まず、分析する。x y 目標画像からu v ソース画像への一般化した逆方向マッピングは、以下のように先に定義した。

#### 【0094】

【数6】

$$\begin{bmatrix} u \\ v \end{bmatrix} = F(x, y) = C \begin{bmatrix} ax + dy + g \\ bx + ey + h \end{bmatrix}$$

ここで、

【0095】

【数7】

$$C = \frac{1}{(cx + fy + 1)}$$

一般化変換のためのヤコビ行列式Jは、xおよびyの非リニア関数である。

【0096】

【数8】

$$J = C^2 \begin{bmatrix} y(af - cd) + a - gc & x(af - cd) - d + gf \\ y(bf - ce) + b - hc & x(bf - ce) - e + hf \end{bmatrix}$$

普通のアンチエイリアシング方法では、ヤコビ行列式は、ソース画像における各ピクセルのフットプリントを決定するのに使用され、異方性フィルタリングにとって必要である。x y ラスタ・スペースにおけるスクリーン・ピクセル間の差は、[1, 0]、[0, 1] 方向において方向性導関数を計算することによってソース画像に投影される。ソース画像スペースにおけるこれらの導関数は、r 1、r 2 と呼ばれ、次のように定義される。

【0097】

【数9】

$$r_1 = J \begin{bmatrix} 1 \\ 0 \end{bmatrix} = C^2 \begin{bmatrix} y(af - cd) + a - gc \\ y(bf - ce) + b - hc \end{bmatrix}$$

および

【0098】

【数10】

$$r_2 = J \begin{bmatrix} 0 \\ 1 \end{bmatrix} = C^2 \begin{bmatrix} x(af - cd) - d + gf \\ x(bf - ce) - e + hf \end{bmatrix}$$

これらのベクトル、 $r_1$ および $r_2$ 、は、フットプリント114を近似する楕円の境界ボックスを定める。代表的には、これらのベクトル116、118は、普通の異方性フィルタリング方法（たとえば、楕円加重平均（EWA）、フットプリント組立体）のために、必要に応じて、すべてのピクセルについて算出される。これは、Cを計算するために1ピクセル当たりもう一回の除算を必要とする。本発明によれば、以下に説明するように、フットプリントを決定するためのより正確な方法が与えられる。

【0099】

ヤコビ行列式が非リニア・マッピングのリニア近似であるから、それはより正確であり、したがって、ソース画像スペースにおける隣接したサンプルまでの距離を取り出すことによってフットプリントを計算するのに好ましい。隣接したサンプルの投影が既に計算されているので、本発明のこの方法は、付加的な除算をなんら必要としない。

【0100】

平行・保存（PP）走査方向は、ヤコビ行列式を計算するのに、より大きい干渉性に備え、除算を与えない。PP走査オーダーにおける各ピクセルについて、フットプリントは、好ましくは、 $r_1'$ 、 $r_2'$ によって定義される。PPスキャンラインに沿った方向 $[1, k]$ における方向性導関数 $r_1'$ は、

【0101】

【数11】

$$r_1' = \nabla_{[1,k]} F = J \begin{bmatrix} 1 \\ k \end{bmatrix} = \dot{C}^2 \begin{bmatrix} af - cd \\ bf - ce \end{bmatrix}$$

であり、そして、 $y=kx+B$ であるから、すべてのPPスキャンラインについて  $C=1/(Bf+1)$  は一定であり、したがって、 $r_1'$  もすべてのPPスキャンラインについて一定である。本発明の方法は、この事実を利用して、好ましくは、除算なしで、ソース画像座標を1つのスキャンラインに沿って増分させる。 $y$ 方向  $[0, 1]$  における方向性導関数  $r_2'$  の値は、

【0102】

【数12】

$$r_2' = \nabla_{[0,1]} F = r_2$$

である。ここで、 $x$ の関数であり、したがって、スキャンラインに沿って増分させることができるので、 $r_2'$  がスキャンラインに沿って線形に変化することは明らかである。この特殊な走査方向は、ソース画像座標およびピクセル・フットプリントを簡単かつ能率的に計算するのを可能にする。

【0103】

すべてのフットプリントおよびソース・ピクセル座標情報を能率的に計算した後、正しい異方性フィルタリングを、当業者にとって公知の標準方法、たとえば、GreeneおよびHeckbertの楕円加重平均(EWA)またはシリング等のフットプリント組立体を使用して実施することができる。これらの普通のアゴリズムは、たとえば、テキストCreating Raster Omnimax Images from Multiple Perspective Views Using the Elliptical Weighted Average Filter, by N. Greene and P. S. Heckbert, IEEE Computer Graphics and Applications, 6 (6) :21-27, June 1986に記載されている。しかしながら、先に指摘したように、楕円フッ

トプリント近似値でさえ不正確であるから、これらの普通のフィルタリング方法は好ましくない。さらに、このような従来技術方法は、冗長なサンプリングを行う（すなわち、複数回、各ソース・ピクセルにアクセスする）ことになる。たとえば、1.0ソース・ピクセルのフットプリント半径を持つ円形フィルタ領域の場合、各ソース・ピクセルは、平均 $\pi$ 回サンプリングされる。本発明の順方向マッピング技術を使用することによって、冗長なメモリ・アクセスが本質的に除去され、したがって、 $\pi$ の因数だけメモリ・バンド幅を下げることができる。好ましくは、本発明は、すべてのソース・ピクセルが、ピクセル読み込みテンプレート・オーダーで一度読み込まれ、次いで、フィルタ・カーネルで目標画像上へスプラットされる順方向マッピング技術を提供する。

#### 【0104】

図13に示すように、各ソース・ピクセル124は、その最も近い隣接の目標サンプル126の各々に対する $\Delta x$ 120および $\Delta y$ 122を有する。 $\Delta x$ は、好ましくは、スキャンラインに沿ったすべてのサンプルが等距離であるので、増分計算され得る。特殊な走査方向は、各スキャンラインに沿って $\Delta y$ が一定となることをほぼ保証する。ラスタ・グリッド場所は真のスキャンライン128からずれているが、実際の距離は、ピクセル読み込みテンプレート130内に記憶され得る小訂正を加えることによって推定可能であり、好ましくはスキャンラインの中で均一である。フィルタ・カーネルは、好ましくは、一度事前計算され、ルックアップ表(LUT)に記憶される。その後、各ソース・ピクセル124の寄与率が、4つ（またはそれより多くの）最も近い隣接の目標サンプル126についてルックアップ表(LUT)にその $\Delta x$ 、 $\Delta y$ によって割り出されると好ましい。目標サンプル126の数は、使用するフィルタのフットプリントに依存する。そして、好ましくは、4から16のサンプルまで変化してもよい。この方法を使用することで、各ソース・ピクセル124は、好ましくは、メモリから一度正確に呼び出され、次いで、4回（またはそれ以上）、ルックアップ表・エントリによって変調され、目標ピクセルに蓄積される。このようにして、最終的なピクセル値は、すぐ近くのソース・ピクセル124の加重平均となる。この加重平均は、各最終ピクセル強度を標準化するためにフィルタ重みの合計での除算を必要

とする。

#### 【0105】

画像ワーピング（広義には2つの画像（たとえば、ソース画像および目標画像）間にジオメトリック変換として定義され得る）に加えて、三次元（3D）ボリューム変換は、ボリューム・レンダリング、ボリューム・モデリングおよび複数のボリュームの整合で重要な役割を演ずる。すべてのアフィン変換の中で、回転は、一般的に、最も多くの計算時間を費やし、最も複雑と考えられる。したがって、本発明に従ってユニバーサル3Dレンダリング・アーキテクチャを提供する際に、以下に詳しく説明するように、任意の3Dボリューム回転を実施するいくつかのユニークな方法が提起される。本発明のユニバーサル3Dレンダリング・ハードウェアはここに記載した3Dボリューム回転方法なしで使用され得るが、これらの方法またはアルゴリズムは、好ましくは、速度および特徴を向上させるために本発明の装置と一緒に実施され、本発明の装置を最も能率的に利用するようになっている。

#### 【0106】

3Dボリューム回転を実施するユニークな方法を説明する前に、最初に使用する用語のいくつかの基本的な定義を与えることが重要である。当業者によって明らかかなように、画像の行、列に対して、ボリューム内のビームとは、1つの主要座標軸線に沿ったボクセルの行と定義することができる（たとえば、xビームは、x方向におけるボクセルの行である）。ボリュームのスライスとは、主要軸線に対して垂直であるボクセルのプレーンである（たとえば、xスライスは、x軸線に対して垂直なプレーンとして定義される）。

#### 【0107】

ボリューム変換を実施する従来技術方法は、代表的には、多パス・アルゴリズムを利用する。これらのアルゴリズムは、通常、画像変換のために使用される複数パス・アルゴリズムの直接的延長である。3D回転を実施する種々の方法が、提案されており、一般的に、複数の二次元（2D）変換あるいは一次元（1D）変換に3D変換を分解することを伴う。これらの従来技術方法は、Volume Rendering, by R. A. Drebin et al., Computer Graphics (SIGGRAPH '88 Proceedi

ngs) , Vol. 22, pp 65-74, August 1988、Three-Pass Affine Transformations for Volume Rendering, by P. Hanrahan, Computer Graphics (San Diego Workshop on Volume Visualization) , Vol. 24, pp 71-78, November 1990およびFast Rotation of Volume Data on Parallel Architectures, by P. Schröder and J. B. Salem, Visualization '91, pp. 50-57, 1991を含む論文の主題となっている。これらの論文は、すべて、参考資料としてここに援用する。しかしながら、これらの公知の3D変換方法は、代表的には、回転品質が低くおよび／または処理速度が遅くなる。

#### 【0108】

三次元(3D)回転を難しくしている特性の1つは、3D回転が、本来、大域的連絡を必要としており、分散したメモリ・モジュールヘデータを書き戻しながらメモリ・コンテンションを生じさせることにある。しかしながら、シャワー変換が最も近い隣接コネクションを利用するので、本発明のユニークなアーキテクチャによって提供されるように、シャワー変換それ自体は、極めて実行可能なマルチ・パイプライン式ハードウェア実装に役立つ。本発明は、さらに、任意の3D回転を実施する新規な方法、特に、3D回転を異なったタイプのシャワー変換のシーケンスに分解することによる方法を提供する。

#### 【0109】

普通の分解方法を用いる場合、2D回転を3つの一次元(1D)シャワーに分解することができるので、3D回転に対する直接的な拡張では、9つの1Dシャワーを必要とする。しかしながら、本発明によれば、以下に詳しく説明するように、任意の3Dボリューム回転についての好ましいシャワー分解方法が4つ提供される。これらの方法とは、4パス2Dスライス・シャワー、4パス2Dビーム・シャワー、3パス・ビーム・スライス・シャワーおよび2パス3Dビーム・シャワー分解である。スケール作業を導入しないことによって、本発明のアルゴリズムは、サンプリング、フィルタリングおよびそれに伴う画像劣化における複雑化を避けることができる。

#### 【0110】

ここで、当業者であれば、3D回転マトリックスを3つの主要軸線回転、 $R_x$  (



$\phi$ )、 $R_y(\theta)$ 、 $R_z(\alpha)$ の連結として表すことができることは明らかであろう。  
ここで、

【0111】

【数13】

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & \sin \phi \\ 0 & -\sin \phi & \cos \phi \end{bmatrix}$$

$$R_y = \begin{bmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 0 & 0 \\ \sin \theta & 0 & \cos \theta \end{bmatrix}$$

$$R_z = \begin{bmatrix} \cos \alpha & \sin \alpha & 0 \\ -\sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

である。この連結を実施するオーダーは、異なる3D回転マトリックスを生じさせる。全体で3D回転マトリックスの6つの順列がある。たとえば、下層の3D回転マトリックスは、 $R_{3D} = R_x(\phi) R_y(\theta) R_z(\alpha)$ として選択された。個々で、

【0112】

【数14】

$$R_{3D} = \begin{bmatrix} \cos \theta \cos \alpha & \cos \theta \sin \alpha & -\sin \theta \\ \sin \phi \sin \theta \cos \alpha - \cos \phi \sin \alpha & \sin \phi \sin \theta \sin \alpha + \cos \phi \cos \alpha & \sin \phi \cos \theta \\ \cos \phi \sin \theta \cos \alpha + \sin \phi \sin \alpha & \cos \phi \sin \theta \sin \alpha - \sin \phi \cos \alpha & \cos \phi \cos \theta \end{bmatrix}$$

後に続くすべての分解のために、上記の3D回転マトリックス( $R_{3D}$ )が使用される。本発明のユニークな方法と他の普通の方法との間の主要な差の1つは

、本発明においては、シャー・シーケンスを得るために、分解を、複数の2D回転シーケンスよりもむしろ、3D回転マトリックスに直接適用するということである。ここで、本発明に従って実施されるシャー操作の任意のものについて、好ましいハードウェア手段としてバレル・シフタを使用できるが、他の任意の手段（たとえば、対数シフタ等）も同様に考えられることは明らかであろう。

### 【0113】

図14に示すように、本発明の一実施例による、二次元（2D）スライス・シャー回転を実施する方法は、好ましくは、3D回転を一連の2Dスライス・シャーに分解することを含む。2Dスライス・シャーにおいては、ボリューム・スライス（すなわち、主投影軸線に沿った、任意2つの軸線に対して平行なボクセルのプレーン）は、単にそのプレーン内で変位されるだけである。スライスは、任意に任意の主要投影軸線に沿って採用することができる。たとえば、図14はyスライス・シャーを示している。2Dyスライス・シャーは、好ましくは、次のように表現される。

### 【0114】

#### 【数15】

$$\begin{aligned}x &= x + a \cdot y \\ z &= z + b \cdot y\end{aligned}$$

2Dyスライス・シャーは、好ましくは、 $S(xz, y, (a, b))$  として書くことができる。これは、x方向における量a 132だけ、そして、z方向における量b 134だけy軸線に沿ったシャーと解釈される。a、bは好ましくは共に定数であるが、a、bが関数も表すことも考えられる。2Dxスライス・シャー、 $S(yz, x, (c, d))$  と、2Dzスライス・シャー、 $S(xy, z, (e, f))$  も同様に定義される。図14を参照して、実線136で表されるボリュームは、点線138によって定義されたボリュームのシャー結果である。

### 【0115】

直観的に、同じ軸線に沿った連続的なシャーは、整合シャーを生成する。たとえば、

【0116】

【数16】

$$\begin{aligned}
 & S(xz, y, (a, b)) \cdot S(xz, y, (a', b')) \\
 &= \begin{bmatrix} 1 & 0 & 0 \\ a & 1 & b \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ a' & 1 & b' \\ 0 & 0 & 1 \end{bmatrix} \\
 &= \begin{bmatrix} 1 & 0 & 0 \\ a+a' & 1 & b+b' \\ 0 & 0 & 1 \end{bmatrix}
 \end{aligned}$$

一般的な3Dマトリックスを2Dシャー・マトリックスから作り上げるために、シャー・プロダクトは、異なったシャーのプロダクト、すなわち、 $S(xz, x, (c, d))$ 、 $S(xz, y, (a, b))$ 、 $S(xy, z, (e, f))$ に制限することができる。しかしながら、これら3つのシャー・マトリックスのプロダクト・マトリックスは、現在のマトリックスにおける定数1により、なお一般的な形態となることはない。したがって、別のシャー・マトリックスは、好ましくは、連結される。ここで、この最終的なシャーは、第1のシャーと同じスライス・シャーである。これは、以下の6つのシャー・シーケンス順列を生じさせる。

【0117】

【数17】

$$\begin{aligned}
 & S(xz, y, (a, b)) \cdot S(xy, z, (e, f)) \cdot S(yz, x, (c, d)) \cdot S(xz, y, (g, h)) \\
 & S(xz, y, (a, b)) \cdot S(yz, x, (c, d)) \cdot S(xy, z, (e, f)) \cdot S(xz, y, (g, h)) \\
 & S(xy, z, (e, f)) \cdot S(xz, y, (a, b)) \cdot S(yz, x, (c, d)) \cdot S(xy, z, (i, j)) \\
 & S(xy, z, (e, f)) \cdot S(yz, x, (c, d)) \cdot S(xz, y, (a, b)) \cdot S(xy, z, (i, j)) \\
 & S(yz, x, (c, d)) \cdot S(xz, y, (a, b)) \cdot S(xy, z, (e, f)) \cdot S(yz, x, (m, n)) \\
 & S(yz, x, (c, d)) \cdot S(xy, z, (e, f)) \cdot S(xz, y, (a, b)) \cdot S(yz, x, (m, n))
 \end{aligned}$$

シャー・シーケンスの各々について、連続的シャー・マトリックスのプロダクト・マトリックスが、好ましくは、計算され、下層の3D回転マトリックスに等しくなるように設定される。たとえば、第1のシャー・シーケンスについては、上記のように、すなわち、 $S(xz, y, (a, b)) \cdot S(xy, z, (e, f)) \cdot S(yz, x, (c, d)) \cdot S(xz, y, (g, h))$ とする。

【0118】

【数18】

$$R_{3D} = R_x(\phi) R_y(\theta) R_z(\alpha) = \\ S(xz, y, (a, b)) \cdot S(xy, z, (e, f)) \cdot \\ S(yz, x, (c, d)) \cdot S(xz, y, (g, h))$$

上記のマトリックス式は、8つの変数、すなわち、a、b、c、d、e、f、g、hを持つ9つの三角法の式を意味する。8つの変数a～hについてこれら9つの式を解くと、次のような結果が得られる。

【0119】

【数19】

$$a = \frac{\sin\theta \sin\alpha (\cos\theta - \cos\phi) + \sin\phi (\cos\alpha - \cos\theta)}{(\cos\theta)^2 \sin\alpha \sin\phi} \\ b = -\frac{\cos\phi \cos\theta - 1}{\sin\phi \cos\theta} \\ c = \cos\theta \sin\alpha \\ d = \frac{-\sin\phi \sin\theta \cos\alpha + \cos\phi \sin\alpha - \cos\theta \sin\alpha}{\sin\phi \cos\theta} \\ e = \frac{\sin\phi \cos\theta + \cos\phi \sin\theta \sin\alpha - \sin\phi \cos\alpha}{\cos\theta \sin\alpha} \\ f = -\sin\phi \cos\theta \\ g = \frac{\cos\theta \cos\alpha - 1}{\cos\theta \sin\alpha} \\ h = \frac{\cos\theta \cos\alpha - 1}{\cos\theta \sin\alpha}$$

同様にして、上記に残っている5つのスライス・シャー・シーケンスについてのシャー・マトリックスを得ることができる。実際、上記の解を持つスライス・シャー・シーケンスは、最も単純な表現を有し、好ましくは、基本シーケンスと呼ぶ。

【0120】

次に図15を参照して、二次元(2D)ビーム・シャー分解による三次元(3D)回転を実施する方法を以下に説明する。まず、或るビーム・シャーを、他の2つの座標をなんら変化させることなく、主方向において変位されただけであるビームと定義する。たとえば、2D x ビーム・シャーは、好ましくは、

【0121】

【数20】

$$x = x + a \cdot y + b \cdot z$$

と表現する。

【0122】

2D x ビーム・シャーは、好ましくは、 $S(x, yz, (c, d))$ と書くことができ、これは、x方向における量aおよびz方向における量bだけx軸線に沿ったシャーと解釈される。2D y ビーム・シャー、 $S(y, xz, (a, b))$ 、そして、2D z ビーム・シャー、 $S(z, xy, (e, f))$ も同様に定義する。図15は、x ビーム・シャーを示しており、そこにおいて、点線146で表されるボリュームは、実線144によって表されるボリューム位置にシャーされる。

【0123】

二次元(2D)ビーム・シャーは、2Dスライス・シャー以上に有利であり、したがって、好ましい。これは、1つのビームが他の2つの座標を変えずに変位されるからである。したがって、2Dビーム・シャー方法の各パスについて再サンプリングがより簡単になる。これは、リニア補間のみが必要高等である。対照的に、2Dスライス・シャー方法は、より複雑であるバイリニア補間を必要とする。

【0124】

2Dビーム・シャー・マトリックス分解と同様に、一般的な3Dマトリックスを2Dビーム・シャー・マトリックス分解から作り上げるために、シャー・プロダクトは、異なるシャーのプロダクト、すなわち、 $S(x, yz, (c, d))$ 、 $S(y, xz, (a, b))$ 、 $S(z, xy, (e, f))$ に制限することができる。しかしながら、これらの3つのシャー・マトリックスのプロダクト・マトリックスは、マトリックスにおける定数1により、なお一般的に形態とならない。したがって、スライス・シャー方法と同様に、好ましくは、別のシャー・マトリックスを連結する。ここで、この最終的なシャーは、第1シャーと同じビーム・シャーである。これは、以下の6つのシャー・シーケンス順列を生じさせる。

【0125】

【数21】

$$\begin{aligned}
 &S(y, xz, (a, b))S(z, xy, (e, f))S(x, yz, (c, d))S(y, xz, (g, h)) \\
 &S(y, xz, (a, b))S(x, yz, (c, d))S(z, xy, (e, f))S(y, xz, (g, h)) \\
 &S(z, xy, (e, f))S(y, xz, (a, b))S(x, yz, (c, d))S(z, xy, (i, j)) \\
 &S(z, xy, (e, f))S(x, yz, (c, d))S(y, xz, (a, b))S(z, xy, (i, j)) \\
 &S(x, yz, (c, d))S(y, xz, (a, b))S(z, xy, (e, f))S(x, yz, (m, n)) \\
 &S(x, yz, (c, d))S(z, xy, (e, f))S(y, xz, (a, b))S(x, yz, (m, n))
 \end{aligned}$$

上記シャー・シーケンスの各々について、連続的シャー・マトリックスのプロダクト・マトリックスが、好ましくは、計算され、下層の3D回転マトリックスに等しくなるように設定される。たとえば、第1のシャー・シーケンスについては、上記のように、すなわち、 $S(y, xz, (a, b)) \cdot S(z, xy, (e, f)) \cdot S(x, yz, (c, d)) \cdot S(y, xz, (g, h))$ とする。

【0126】

【数22】

$xy, (e, f)) S(x, yz, (c, d)) S(y, xz, (g, h))$ :

$$R_{3D} = R_x(\phi) R_y(\theta) R_z(\alpha) =$$

$$S(y, xz, (a, b)) \cdot S(x, xy, (e, f))$$

$$\cdot S(x, yz(c, d)) \cdot S(y, xz, (g, h))$$

上記のマトリックス式は、8つの変数、すなわち、a、b、c、d、e、f、g、hを持つ9つの三角法の式を意味する。8つの変数a～hについてこれら9つの式を解くと、次のような結果が得られる。

【0127】

【数23】

$$a = \frac{\sin\theta \sin\alpha (\cos\phi - \cos\theta) + \sin\phi (\cos\theta - \cos\alpha)}{\sin\phi \sin\alpha (\cos\theta)^2}$$

$$b = -\frac{\cos\phi \cos\theta - 1}{\sin\phi \cos\theta}$$

$$c = -\cos\theta \sin\alpha$$

$$d = \frac{\cos\theta \sin\alpha + \sin\phi \sin\theta \cos\alpha - \cos\phi \sin\alpha}{\sin\phi \cos\theta}$$

$$e = -\frac{\cos\phi \sin\theta \sin\alpha - \sin\phi \cos\alpha + \sin\phi \cos\theta}{\cos\theta \sin\alpha}$$

$$f = \sin\phi \cos\theta$$

$$g = -\frac{\cos\theta \cos\alpha - 1}{\cos\theta \sin\alpha}$$

$$h = \frac{\sin\phi \sin\theta (\cos\theta - \cos\alpha) + \sin\alpha (\cos\phi - \cos\theta)}{\sin\phi \sin\alpha (\cos\theta)^2}$$

同様にして、上記の残りの5つのビーム・シャー・シーケンスについてのシャー・マトリックスを得ることができる。上記の解を持つビーム・シャー・シーケンスは、好ましくは、基本シーケンスと呼ぶ。

【0128】

次に図16を参照して、本発明による、二次元(2D)ビーム・スライス・シ

チャー分解による三次元（3D）回転を実施する方法を説明する。2Dビーム・スライス・チャーは、好ましくは、1つのプレーン内で変位させられたビームと定義することができる。たとえば、2D x ビーム y スライス・チャーは、好ましくは、

【0129】

【数24】

$$x = x + a \cdot y + g \cdot z$$

$$z = z + b \cdot y$$

と表現される。

【0130】

2D x ビーム y スライス・チャーは、好ましくは、 $S((x, yz, (a, g)), (z, y, b))$ と書くことができ、これは、y方向において量a、z方向において量gだけx軸線に沿ったチャーをy方向において量bだけz軸線に沿ったチャーと組み合わせたものと解釈される。ここで、a、gおよびbは、好ましくは、定数である。本質的に、ビーム・スライス・チャーは、ビーム・チャーとスライス・チャーの組み合わせである。図16は、xビームyスライス・チャー、 $S((x, yz, (a, g)), (z, y, b))$ を示しており、そこにおいては、点線156で表されるボリュームは、実線154によって表されるボリューム位置にチャーされる。

【0131】

一般的な3Dマトリックスを2Dチャー・マトリックス分解から作り上げるために、チャー・プロダクトは、異なったチャーのプロダクト、すなわち、yビームxスライス・チャー $S((y, xz, (c, h)), (z, x, d))$ 、xビームyスライス・チャー $S((x, yz, (a, g)), (z, y, b))$ およびyビーム・チャー $S(y, xz, (I, f))$ に制限するとよい。スライス・チャー、ビーム・チャー方法の場合と同様に、6つのビーム・スライス・チャー・シーケンス順列があることは了解されたい。

【0132】



シャー・シーケンスの各々について、連続的シャー・マトリックスのプロダクト・マトリックスが、好ましくは、計算され、下層の3D回転マトリックスに等しくなるように設定される。たとえば、上記の第1のシャー・シーケンスについては、 $S((y, xz, (c, h)), (z, x, d)) S(x, yz, (a, g)), (z, y, b) S(y, xz, (I, f))$ とする。

【0133】

【数25】

$$(z, x, d)) S((x, yz, (a, g)), (z, y, b)) S(y, xz, (I, f)):$$

$$R_{3D} = R_x(\phi) R_y(\theta) R_z(\alpha) = \\ S((y, xz, (c, h)), (z, x, d)) \cdot S(x, yz, (a, g)), (z, y, b)) \\ \cdot S(y, xz, (I, f))$$

上記のマトリックス式は、8つの変数、すなわち、a、b、c、d、f、g、h、Iを持つ9つの三角法の式を意味する。8つの変数についてこれら9つの式を解くと、次のような結果が得られる。

【0134】

【数26】

$$\begin{aligned} a &= \sin\phi \sin\theta \cos\alpha - \cos\phi \sin\alpha \\ b &= \sin\phi \cos\theta \\ c &= \frac{\sin\phi(\cos\theta - \cos\alpha) + \sin\theta \sin\alpha(\cos\phi - \cos\theta)}{\sin\phi(\cos\theta)^2 \sin\alpha} \\ d &= \frac{\sin\phi \cos\alpha - \cos\phi \sin\theta \sin\alpha - \sin\phi \cos\theta}{\cos\theta \sin\alpha} \\ f &= \frac{\sin\phi \sin\theta(\cos\theta - \cos\alpha) + \sin\alpha(\cos\phi - \cos\theta)}{\sin\phi(\cos\theta)^2 \sin\alpha} \\ g &= \frac{\cos\theta \sin\alpha + \sin\phi \sin\theta \cos\alpha - \cos\phi \sin\alpha}{\sin\phi \cos\theta} \\ h &= \frac{\cos\phi \cos\theta - 1}{\sin\phi \cos\theta} \\ i &= -\frac{\cos\theta \cos\alpha - 1}{\cos\theta \sin\alpha} \end{aligned}$$

残りの5つのビーム・シャー・シーケンスについてのシャー・マトリックスを同様にして得ることができることは明らかである。

#### 【0135】

図17は、本発明に従って、3Dビーム・シャー分解を使用して任意の三次元(3D)回転を実施する第4の方法を示している。上記のビーム・スライス・シャー分解方法で使用した連続的シャー・マトリックスのプロダクト・マトリックス(すなわち、 $S(y, xz, (c, h))$ 、 $(z, x, d) \cdot S(x, yz, (a, g))$ 、 $(z, y, b) \cdot S(y, xz, (l, f))$ )をさらに点検することによって、2Dシャーの最初のペア、最後のペアを合併することができる。これは、各ペアには共通のビームがあるからである。たとえば、xビームは、第1ペアのyスライス、zスライスのシャーの共通ビームである。したがって、シャーの数は、3Dビーム・シャーの新定義を導入することによって2まで減らすことができる。

#### 【0136】

図17は、3Dxビーム・シャーを示している。これは、2つの連続した2Dスライス・シャー $S(xz, y, (a, b))$   $S(xy, z, (e, f))$ の連結に等しい。ここで、他に2つの3Dビーム・シャーがある、すなわち、 $S(yz, x, (c, d))$   $S(xz, y, (a, b))$ と表される3Dzビーム・シャーと、 $S(yz, x, (c, d))$   $S(xy, z, (e, f))$ と表される3Dyビーム・シャーとがあることは了解されたい。すべての3Dビーム・シャーは、好ましくは、たった1つの主ビームだけを含む。図17を参照して、マークを付けたxビーム158(暗い陰影のビーム)は、好ましくは、矢印をたどる新しい3D場所に並進せられる。明るい陰影を付けたビーム158'は、シャー分解を2つの連続的2Dスライス・シャーと解釈された場合の中間位置を示す。

#### 【0137】

3つの3Dビーム・シャーは、好ましくは、 $SH_{3Dx}$ 、 $SH_{3Dy}$  および  $SH_{3Dz}$  と示すとよい。ここで、ここに記載した本発明の方法を用いた場合、任意の3D回転を、ほんの2つの連続3Dビーム・シャーに分解できる。基本分解シーケンスは、以下のように2Dスライス・シャー・シーケンスから直接得ることができる。

【0138】

【数27】

$$R_{3D} = SH_{3D_x} \cdot SH_{3D_z}$$

ここで、

【0139】

【数28】

$$SH_{3D_x} = S(xz, y, (a, b)) \cdot S(xy, z, (e, f))$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ a+be & 1+bf & b \\ e & f & 1 \end{bmatrix}$$

$$SH_{3D_z} = S(yz, x, (c, d)) \cdot S(xz, y, (g, h))$$

$$= \begin{bmatrix} 1+cg & c & d+ch \\ g & 1 & h \\ 0 & 0 & 1 \end{bmatrix}$$

ここに説明した本発明の3Dビーム・シャープ分解方法を使用すると、任意の3D回転は、好ましくは、2つの主要ビーム変換しか伴わないのに対して、普通の分解方法は、3つ（たとえば、Hanrahanの分解）を必要とする。本発明の3Dビーム・シャープ方法によれば、第1のパスはxビームのみを含み、第2のパスはzビームのみを含む。第1のシャープ・パスの終りまでに、ビームのすべてのボクセルは、好ましくは、同じオフセットを有する。 $N^3$ 個のボリュームについて $N^2$ 個のビームがあるので、異なったオフセット値は $N^2$ 個だけある。したがって、 $N^2$ 個のビームについてのオフセット値は、第1パスの終りで記憶することができると共に、最も近い隣接の一体位置にボクセルを記憶することができる。

【0140】

多パス・アルゴリズムが使用されるとき、選ばれた再サンプリング技術は、高

品質を達成するのに重要である。直観的に、連続するシャー変換がグリッド・ポイントを離れるようにボクセルを移動させることができるので、再サンプリングが各パスについて必要である。しかしながら、連続回転をボリュームに適用した場合、多数回の再サンプリングに固有の問題の1つは、ボリューム品質の急速な劣化にある。したがって、ボリュームを一度だけサンプリングすることが望ましい。

#### 【0141】

したがって、本発明の好ましい方法は、ボリュームの1パス式再サンプリングを達成する。本質的に、本発明の方法は、サンプリングしたボリュームを事前計算し、次に、各シャー・パスにおいてゼロ・オーダー（たとえば、最も近い隣接の）補間を使用する。これは、大域的連絡を必要とする公知の従来技術方法（たとえば、Wittenbrink、Somaniの順列ワーピング）と異なる。

#### 【0142】

或るオリジナルのボリューム（ソース・ボリューム）と所望の回転したボリューム（目標ボリューム）とを与えた場合、本発明の方法は、好ましくは、まず、ソース・ボクセルと目標ボクセルとの間の1対1の対応関係を構築する。この1対1のマッピングは、各シャーがゼロ・オーダー補間を使用した1対1変換であるから、本発明のマルチパス・シャー分解によって保証される。一連の1対1マッピングの連結は、1対1のままである。ひとたびこの1対1の対応関係が確立されたならば、本発明の方法は、好ましくは、各ソース・ボクセルについて、それに対応する目標ボクセルを計算し、それをソース・ボクセル位置に記憶する。この手順中、大域的連絡はまったく不要である。すなわち、再サンプリングが、局所的ボクセルについての補間によって実施される。各目標ボクセルのサンプリング位置は、好ましくは、逆方向回転変換を使用して計算する。

#### 【0143】

すべての目標ボクセルについて値を得た後、本発明の方法は、好ましくは、目標ボリュームにおける宛先に目標ボクセルをシャッフルする。直観的に、これは大域的連絡を含む。しかしながら、並列実行のために行うには大域的連絡は高価である。したがって、本発明による方法は、好ましくは、最も近い隣接配置方式

と共に複数のシャーを使用し、このボクセル・シャッフリングを達成する。シャーが規則的な非対立変換であるから、各パスは、大域的連絡を利用する場合よりも能率的に実施され得る。ここに説明した本発明の3Dビーム・シャー分解方法を使用することで、実質的に大域的連絡と同じ効果を達成するのに、規則的な局所的連絡のパスを最低2回行うだけでよい。

#### 【0144】

ここで、3Dビーム・シャーにおけるビームのオーバーラップを避けることに注意しなければならないことは明らかである。たとえば、ここで、上記の3D x ビーム・シャー式を考える。各xビームを保存するとき（すなわち、xビームが3D x ビーム・シャーの後に剛性のままとなる）、いくつかのxビームが互いにオーバーラップする可能性がある。必要な1対1マッピングを維持するために、上述したように、3Dビーム・シャーが2つの2Dスライス・シャーの連結であることを想起されたい。ゼロ・オーダー補間を使用するとき、2Dスライス・シャーは1対1マッピングを維持する。したがって、解として、本発明の方法は、好ましくは、2つの連続した2Dスライス・シャーのオーダーと同じオーダーを用いて宛先座標を計算するが、連絡は一度だけ行われるのが好ましい。3D x ビーム・シャーの場合、3Dシャー・マトリックス（上記）を用いてx座標を直接的に算出しながら、各ビームのy、z座標を、次のように計算すると好ましい。

#### 【0145】

#### 【数29】

$$z' = \text{round}(z + b \cdot y)$$

$$y' = \text{round}(y + f \cdot z')$$

ここで、round(w)は、最も近い整数に対する丸め関数wである。座標(y'、z')は、最も近い隣接のストレージについてのビーム全体の一体座量を決定する。こうして、オーバーラップが生じることはない。

#### 【0146】

本発明の別の形態によれば、強化ボリューム処理を実施するいくつかのユニー

クな方法を以下に詳しく説明する。

【0147】

透視投影は、特にレイ・キャスティングを実行するときに、固有の問題を提起する。平行投影の場合、ボリューム・データセットを通してキャスティングされるサイト・レイは、下層のボリューム・データについて一定のサンプリング・レートを維持する。必要な品質の出力画像を創作するようにこのサンプリング・レートをセットするのは簡単である。しかしながら、透視投影の場合、レイは、このような連続して均一なサンプリング・レートを維持しない。その代わりに、レイは、前から後へボリュームを移動させるにつれて発散する。図18a、18bに示すように、これは、下層のボリュームに不均一なサンプリングを創り出す。

【0148】

次に図18a、18bを参照して、普通のビーム・キャスティング・アルゴリズムは、一般的に、2つの方法のうちの1つによって、透視投影からのレイ発散を取り扱う。第1の方法は、アンダーサンプリング（図18A）であり、そこにおいては、レイ160が所定の視点162からキャスティングされ、その結果、ボリューム164のフロントでのサンプリング・レートが所望の画質に対して適切なものとなる。しかしながら、透視レイ発散のために、下層のボリューム・データセットは、アンダーサンプリングされる。これは、ボクセルの領域がサンプリングされないままであるボリューム166のリアに「ホール」を創り出すことによって厳しいエイリアシングとなる可能性がある。第2の方法は、オーバーサンプリングであり（図18B）。そこにおいて、レイ160が所定の視点162からキャスティングされ、その結果、ボリューム・データセット166のリアでのサンプリング・レートが、所望の画質にとって適切なものとなる。この方法では、第1の方法のエイリアシングを回避することができるが、ボリュームはフロント164において急進的にオーバーサンプリングされる可能性がある。ボリューム164のフロントにおける効率の悪いオーバーサンプリングは、この方法のランタイムを劇的に増大させる。レイ160は、アンダーサンプリングとオーバーサンプリングの間のサンプリング・レートでキャスティングされ得る。これは、オーバーサンプリングの画質とアンダーサンプリングのレンダリング速度との

間のトレードオフとなる。

【0149】

多くの従来の像形成アーキテクチャは、透視投影を実施しようという試みさえしていない。他のアーキテクチャは、所定の視点からの発散サイト・レイをキャストリングすることによって、透視投影を処理していた。これは、一時的なエイリアシングで画像を生成し、真のリアルタイム・フレーム率（すなわち、30ヘルツ）を達成せず、本発明のスライス・オーダー方法よりもかなり複雑である。

【0150】

レイ分割方法は、適応スーパーサンプリングの概念を適用し、均一なレイ密度を維持する。この方法では、隣接したレイが或る種の所定閾値を越えて発散するとき、レイを2つのチャイルド・レイに分割する。最近になって、視点からの距離に基づいて視野切頭体を領域に分割する方法が提案された。その結果、各領域のレイ密度が下層ボリューム解像度に近くなるのである。その後、この方法では、サブ画像に各領域を投影し、テクスチャ・マッピング・ハードウェアを使用してフレーム・バッファにこれらのサブ画像を合成する。実際、この技術は、連続的なレイを1つの領域を通してキャストリングし、次いで、指定境界のところで、それらを新しいセットの連続レイに分割する。しかしながら、これは、領域間の潜在的な、望ましくない不連続性を創り出す。

【0151】

本発明の一形態による、ER透視（指数的な領域透視）と呼ばれる、均一な規則的データセットの透視投影を実施するための方法は、好ましくは、下層のボリュームを適切にサンプリングし、それによって、普通のボリューム・レンダリング・システム及び方法に固有の上記問題がほぼ除去される。ER透視アルゴリズムは、アンダーサンプリング、オーバーサンプリングの両方の望ましい特性を組み合わせ、アンダーサンプリング方法のオーダーについてのランタイムを与えながら、オーバーサンプリングと関連した極めて良好なアンチエイリアシング特性を提供する。さらに、このアルゴリズムは、好ましくは、ボリューム・データセットのすべての可視ボクセルについて少なくとも1つのサンプルを創り出す。ER透視は、公知の透視投影方法と比較して同等あるいはそれより良好な画質を維

持しながらスライス・オーダー・ボクセル・アクセスを利用することによって従来の技術より以上のランタイム利益を獲得する。

#### 【0152】

図19は、本発明によるER透視アルゴリズムの2D頂面図である。図19に示すように、ER透視アルゴリズムは、好ましくは、所定の視点172からの主要投影軸線（たとえば、z軸）に沿った指数的に増大する距離に基づいて視野切頭体168を複数の領域に分割することによって機能する。好ましくは、連続したサイト・レイ174がボリューム・データセットを通して後から前（または前から後）へ視点172からキャストイングされ、ひとたびレイ174が互いに接近しすぎ（あるいは離れすぎた）ならば、これらのレイ174を合併（あるいは、分割）する。ER透視アルゴリズムの動作が前から後へのレイ・キャストイングと比較して後から前に付いて類似しているので、ER透視アルゴリズムの残りの説明は、合併を伴う後から前へのレイ・キャストイングのより直観的なケースに限定する。これらの差異は、それらが意味を持つときに指摘する。

#### 【0153】

ER透視アルゴリズムは、好ましくは、領域境界170を使用する。これは、指数領域を定め、サイト・レイ174が好ましくは合併される場所にマークを付ける。領域を定義して、境界170ですべてのレイ174を合併することによって、このアルゴリズムは、局所的近傍状況よりもむしろ大域的ジオメトリに依存する或る規則的なレイ合併パターンを提供する。図20Aは、特に、ベースプレーン・ピクセルBに対する寄与率について領域境界170でサイト・レイを合併することをより明瞭に示している。図20Aを参照して、奇数のレイ174が、好ましくは、合併され、その結果、生じたレイ174'は、本質的に、先の中心レイの正確な連続レイとなり、したがって、領域境界170に存在する潜在的な不連続性を排除する。これは、公知従来方法以上の、本発明方法の重要な利点の1つである。さらに、このアルゴリズムは、最適状態でボリュームをサンプリングするときに達成されるフィルタリングを特徴づけることによって限定され得る。

#### 【0154】



好ましいフィルタ方式の一例が、図20Bに示してある。図20Bを参照して、バートレット・ウィンドウ（すなわち、リニア補間、三角形フィルタ）を使用すると好ましい。各領域境界170で効率的な局所バートレット・ウィンドウをカスケードリングすることは、各ベースプレーン・ピクセル（図20A参照）についての単一の大きなバートレット・フィルタでレイ174を再サンプリングすることとほぼ同等である。好ましいフィルタ重み175のグラフ図が、ベースプレーン・ピクセル（たとえば、ピクセルA、B、C）に対する寄与率について示してある。

#### 【0155】

このアルゴリズムの基本的なサンプリング・レートは、所望の画質に従って所定の値にセットすることができる。基本サンプリング・レートは、下層のボリューム解像度と比較した最小限のビーム密度である。本発明のER透視方法は実質的に任意のサンプリング・レートをサポートするけれども、1ボクセル当たり少なくとも1つのレイのサンプリング・レートが好ましい。このアルゴリズムは、基本サンプリング・レートの一倍と二倍の間にレイ密度を維持するという利点を有する。このことは、ボクセルがボリューム・データセットの背後で失われることがないことを保証し、二倍のスーパーサンプリングで実行された総仕事量について上限を置く。

#### 【0156】

本発明がスライス・オーダー処理を利用するので、ボリューム・データセットは、視野方向に対して最も垂直であるボリュームのベースプレーン上へ投影される。次に、ベースプレーン画像は、普通の方法（たとえば、シャー・ワープあるいは従来のCube-4アーキテクチャと同じ方法）で最終的な画像プレーン上へワープされる。

#### 【0157】

本発明のER透視方法は、上記のCube-5アーキテクチャで実施するのに理想的である。具体的には、このアルゴリズムは、好ましくは、処理要素間の最も近い隣接連絡を必要とするだけである。処理要素の一次元アレイ上のボクセル行を処理しながら、このアルゴリズムは、処理要素がそれらの直ぐ左隣、右隣の

処理要素と連絡することを必要とするだけである。Cube-5レンダリング・パイプラインは、同様に、最も近い隣接連絡をサポートする。

#### 【0158】

本発明のER透視アルゴリズムは、好ましくは、3つの主要軸線の1つに沿ったスライス・オーダー処理を使用する。したがって、ER透視アルゴリズムにおける領域は、主要投影軸線に沿ったスライスのスラブとして定義される。本発明のER透視方法の好ましい実施例においては、ボリューム・データセットは、 $z$ 軸に対して垂直なスライスに沿って投影される。 $z$ 軸のみに沿った投影に本発明の方法を限定するつもりはないが、座標系をフリップさせ、ジオメトリを回転させ得ることは了解されたい。本アルゴリズムは、図7に示すように、視点86からボリューム・データセット80のフロントまでの、 $z$ 軸に沿った距離を測定し、決定する( $e_z$ )ことによって進行する。次いで、この距離と同程度の多くの $z$ スライスからなる第1の領域92が創り出される。第1領域92の後に続く各領域は、好ましくは、その前の領域の二倍深い。

#### 【0159】

高品質スーパーサンプリングと組み合わせたとき、第1領域は、最終画像の1ピクセル当たり1つのレイを放射するとき、正確に、領域の終りで1ボクセル当たり1つのレイを持つのに必要ほど大きい。したがって、二倍より高いスーパーサンプリングが、高品質クローズアップ画面をレンダリングするのにボリュームのフロントに必要となるかも知れない。

#### 【0160】

図19に例示したように、視点172が、たとえば、ボリュームのフロント（すなわち、 $z=3$ 領域境界）から3つのボクセル単位である場合、第1領域176は、好ましくは、3ボクセル単位厚となり、次の領域は、6ボクセル単位厚、以下同様となる。一般に、 $I$ 番目の領域は、好ましくは、 $e_z \cdot 2^I$  スライス厚である。ここで、 $e_z$ は、視点172からボリュームのフロント（図19参照）までの距離である。領域をこのように定義することで、領域の任意のものを通してキャスティングされた任意2つの透視レイ174が、前部境界（すなわち、 $z=3$ 境界）のところにあるときに二倍ほど後部境界（すなわち、 $z=24$ 境界）で離れ

ているという所望の効果を生み出す。これは、2つのレイ174間の距離が、第1領域176を横切って1単位から2単位まで成長し、次いで、4単位、最終的に、最終領域のところで8単位まで成長するように、図19に示してある。さらに、領域境界170が大域的ジオメトリに依存しているので、このレイ・キャスティング・アルゴリズムの効率は、各ディメンジョンでボリューム解像度の一倍から二倍にレイ密度を保つための機構を設けることによって、最大となる。これは、また、透視レイがキャスティングされるときにデータのフィルタリングが制御されるようにも規則的なトポロジを創り出す。

#### 【0161】

指数的距離のところに境界を持つ領域を設けることで、領域のバックのところよりもフロントで二倍高いレイ密度を生み出す。したがって、領域境界と交差するときにレイ密度を調整する機構を設けると好ましい。各レイが領域のリアでボクセル座標について出発すると好ましいので、領域のフロントでは、各ディメンジョンにおけるすべての第2レイは、好ましくは、ボクセル座標と直接一致する。残りのレイは、好ましくは、2つのボクセル位置間の途中で領域と交差する。この決定的なレイ・パターンを持つレイ密度をダウンサンプリングするために、二次元(2D)バートレット・フィルタ(テントまたは三角形フィルタとしても知られている)を、各ディメンジョンで±1ボクセルの範囲で、使用すると好ましい。各領域のフロントでのレイ密度がボクセル密度の二倍なので、この3×3ボクセル近傍は5×5レイと交差する。次に図21を参照して、エッジ178の各々が、重みゼロであるから、3×3の隣接したレイ180のみを使用して、レイ密度をダウンサンプリングするのにフィルタを適用することができる。これは、隣接したレイを効果的に合併する。バートレット・フィルタは、最終的な画像で生み出す付加的な品質のための簡単なボックス・フィルタよりも好ましい。前から後への処理の場合は、レイは合併するかわりに分割される。ここで、レイのバイリニア補間が実施されて、他のレイ間で開始する新しいレイを生成する。ここで、±1のサイズのバートレット・フィルタが、バイリニア補間作業の逆のものであることは言うておかなければならない。

#### 【0162】

図22は、視点196がボリュームのフロントで（すなわち、ベースプレーン198から）3ボクセル単位であるとき、サイト・レイ186が $7^3$ 個のボリューム192を通してどのように移動するか2D例を示している。サンプリング・レートが1スライス当たり7と14の間に留まっており、そして、レイ186が領域を通して後から前に移動するにつれて増大することによって注意されたい。 $N^3$  ボリュームについてレイ密度再サンプリング・ステージの数は、それが $N^3$  ボリュームにおける最大領域数であるから、ので、のためのステージを再サンプリングしているレイ密度の数は、 $\log_2 N$ によって制限される。最終的な画像ワープが生じたとき、ベースプレーン198に示される最終再サンプリング・ステップが実施されると好ましい。

#### 【0163】

図22に示すように、ボリューム・データセット182のリアは、必ずしも常に領域境界184と一致するわけではない。しかしながら、レイ186が領域境界184のすべてで正確なボクセル座標188上に存在することが好ましいので、レイ186は、ボリューム・データセット192を囲む最終領域（陰影付き領域）のリアにおいてグリッド座標190上で発祥するのが好ましい。したがって、ボクセル座標およびレイ・サンプル場所194は、ボリューム182のリアに適合してはならない。これは、前述の境界条件を提供するだけでなく、視点196がボクセル単位距離より小さく動かされるときに一時的なアンチエイリアシングを助けもする。それは、レイ186がボクセルに対して同じ位置から発祥し続けるからである。

#### 【0164】

図23は、本発明の一形態に従って、ボリュームの後から前へのER透視投影を実施する好ましい方法を示しているが、このER透視方法の他の実施例も考えられる。上記のように、まず、目または視点からベースプレーンまでの距離を、好ましくは、決定する（ボクセル単位）。この視点位置を使用して、指數的領域境界を創り出す。次に、好ましくは、十分な領域を確立し、ボリューム・データセットを完全に取り囲む。ボリューム・レンダリングを実施するために、アルゴリズムは、後から前へ各領域を通してループし、ノーマル・レイ・キャストイン

グを計算するが、スライス・オーダー様式では、合成バッファに部分的に計算されたレイを記憶する。領域間（すなわち、領域境界）では、上記したように、合成バッファのレイ密度再サンプリングが実施されると好ましい。次に、ベースプレーン画像を、ディスプレイのために最終的な画像プレーン上へワープさせる。

#### 【0165】

従来技術において知られている適応レイ密度透視方法の場合、一般的に、レイが不規則なパターンを使用して合併されるときに行われるフィルタ機能を決定することは困難である。しかしながら、本発明のER透視方法は、好ましくは、フィルタ動作のための規則的な境界を使用し、これらの境界内に正確にレイを配置するので、局所的パートレット・フィルタのカスケーディングによって行われる効果的フィルタを計算することはより容易である。これは、本発明のER透視アルゴリズムの重要な利点である。さらに、本発明の境界およびフィルタは、好ましくは、個別のデータについての普通の連続したフィルタリングと通常関連する低画質を克服するように選ばれている。

#### 【0166】

たとえば、ここで、図24に示すように、ボリウムの前方2ボクセル単位にある視点と共に7スライス分深いボリウムの透視投影を考える。本発明のER透視方法を使用することで、領域を通してキャストされたレイ200は、領域の後部のところで1ボクセル単位離れる。しかしながら、レイが領域境界202に達したとき、これらのレイは、好ましくは、局所的パートレット・フィルタを使用して濾過される。パートレット・フィルタ（1次に単純化された）は、サイズ $2n+1$ のカーネルについて以下の重みを含み、出力が入力と同じスカラー範囲を有するように標準化される。

#### 【0167】

#### 【数30】

$$0, \frac{1}{n^2}, \frac{2}{n^2}, \dots, \frac{n-1}{n^2}, \frac{n}{n^2}, \frac{n-1}{n^2}, \dots, \frac{2}{n^2}, \frac{1}{n^2}, 0$$

領域境界での二次元画像の場合、本発明は、好ましくは、2つの主要方向に2つの一次元バートレット・フィルタを巻き込むことによって、二次元バートレット・フィルタを使用する。ER透視アルゴリズムは、好ましくは、常にレイを再サンプリングしてオリジナル密度の半分を有する。サイズ±2のフィルタを使用することで、レイ (n=2) は、5×5のフィルタ・カーネルまたは一ディメンジョンについて以下のたった5つ重みを創り出す。

【0168】

【数31】

$$0, \frac{1}{4}, \frac{2}{4}, \frac{1}{4}, 0$$

たとえば、図24に示すように、サンプルa、b、c、d、eの、場所oで領域2から領域1へ変化する部分合成レイに対する寄与率を考える。

【0169】

【数32】

$$o = \frac{1}{4} b + \frac{2}{4} c + \frac{1}{4} d$$

同様に、場所pおよびqでの部分的なレイを次のように計算する。

【0170】

【数33】

$$p = \frac{1}{4} d + \frac{2}{4} e + \frac{1}{4} f$$

$$q = \frac{1}{4} f + \frac{2}{4} g + \frac{1}{4} h$$

ピクセルAについての最終フィルタにおける重みがゼロなので、部分レイn、r

についての式は省略されている。ER透視アルゴリズムを継続することで、再サンプリングされた部分レイ  $n$ 、 $o$ 、 $p$ 、 $q$ 、 $r$  は、好ましくは、領域1を通じてキャストイングされ、そこにおいて、局所的バートレット・フィルタによって再度濾過される。 $n$ 、 $o$ 、 $p$ 、 $q$ 、 $r$  の、ピクセルAに対する標準化寄与率は、

【0171】

【数34】

$$A = \frac{1}{4} o + \frac{2}{4} p + \frac{1}{4} q$$

$o$ 、 $p$ 、 $q$ の値を代入して、

【0172】

【数35】

$$A = \frac{1}{16} b + \frac{2}{16} c + \frac{3}{16} d + \frac{4}{16} e + \frac{3}{16} f + \frac{2}{16} g + \frac{1}{16} h$$

ここで、この式が、9つの値のカーネル・サイズ ( $n=4$ ) を持つバートレット・フィルタと同じ重み（すなわち、係数）を含むことは了解されたい。これは、同じフィルタ重みを持つピクセルBについて繰り返すことができる。前から後への処理の場合、同様の分析を利用して、アルゴリズムの性能およびバイリニア補間の連続適用の結果を示すことができる。

【0173】

スライスの各サンプルは、好ましくは、同じ領域にある他の任意のサンプルと同じ量、最終画像に寄与する（カラー・マッピングおよび合成のようなサンプルに対する他のすべての動作が等しいと仮定する）。たとえば、サンプル  $e$  の値は、局所的バートレット・フィルタのカスケディング後の  $1/4$  の効果的な重みをもってピクセルAに寄与する。同様に、サンプル  $i$  は、 $1/4$  の有効重みでピクセルBに寄与する。サンプル  $f$  は、全体で  $1/4$  について  $3/16$  の重みでピクセルAに寄与し、そして、 $1/16$  の重みでピクセルBに寄与する。これは、

サンプルgおよびhについて繰り返すことができる。それぞれ、サンプルeの左、サンプルIの右のサンプルは、それぞれ、ピクセルAの左のピクセル、ピクセルBの右のピクセルに部分的に寄与し、最終画像に対するこれらの寄与率の合計もまた $1/4$ となる。実際、この領域にあるすべてのサンプルは、同じ重みを有する。この領域がボリュームにおける第2の領域であるから、重みは $1/4$ である。ボリュームの第1領域について、あらゆるサンプルは、好ましくは、 $1/2$ の重みを有する。これは、この領域の最終画像ピクセル当たり2つのレイがあることを実現することによって修正可能である。第2領域等における最終画像ピクセル当たりのレイは4つである。したがって、最終画像に向かう各ピクセルの寄与率を決定する重みは、(画像ピクセル) / (このスライス内のサンプル) となる。

#### 【0174】

本発明のER透視方法がスライス・オーダー処理を実施するので、全計算量は、各スライスに実施される仕事量を算出することによって分析することができる。各サンプルについて行われた仕事が同じものであると仮定するならば、処理されるサンプルのカウント数を、作業負荷の比較として使うことができる。たとえば、オーバーサンプリング方法(図18B参照)においては、領域境界上で正確に終わっているボリュームのリアにあるサンプル数は、 $N^2$ である。フロント・スライスでは、サンプルのカウント数は、視点のジオメトリに依存する。特に、類似した三角形を使用し、視点からボリュームのフロントまでの距離として $e_z$ と定義するならば、取り出されるサンプルの数は、

#### 【0175】

#### 【数36】

$$\left( \frac{N^2 + N \cdot e_z}{e_z} \right)^2$$

これは、ボリューム・データセットを通して任意のスライスsについて

#### 【0176】



【数37】

$$\left( \frac{N^2 + N \cdot e_i}{e_i + s} \right)^2$$

に一般化することができる。したがって、オーバーサンプリング方法によって処理されるサンプルの総カウント数は、

【0177】

【数38】

$$\sum_{i=0}^N \left( \frac{N^2 + N \cdot e_i}{e_i + s} \right)^2$$

となる。同様に、アンダーサンプリング方法（図18A参照）は、以下の仕事量を実施するように示すことができる。

【0178】

【数39】

$$\sum_{i=0}^N \left( \frac{N \cdot e_i}{e_i + s} \right)^2$$

本発明のER透視アルゴリズムの場合、分析はより複雑となる。視野ジオメトリに依存して、

【0179】

【数40】

$$\log \left( \frac{N + e_i}{e_i} \right) - 1$$

領域が創り出される。先に示したように、これらの領域の各々は、好ましくは、 $e_z \cdot 2^1$  スライスを含む。ここで再び、類似三角形のジオメトリック原理を使用するならば、本発明のER透視アルゴリズムは、以下の数のサンプルを処理する。

【0180】

【数41】

$$\sum_{reg=0}^{\log\left(\frac{N+e_z}{e_z}\right)-1} e_z \cdot 2^{reg} \left( \frac{N * (e_z * 2^{reg} - e_z + s)}{e_z * 2^{reg} - e_z} \right)^2$$

この式は、以下の上限を有する。

【0181】

【数42】

$$\sum_{s=0}^N (2N)^2$$

そして、以下の下限を有する。

【0182】

【数43】

$$\sum_{s=0}^N N^2$$

オーバーサンプリング方法（上記のもの）によって処理されるサンプルの総カウント数についての式を点検すると、視点がボリュームに非常に近いとき、オーバーサンプリング方法がフロント・スライスに  $O(N^1)$  作業を実行することがわかる。視点がボリュームのフロントにより接近するように移動するにつれて、

オーバーサンプリング・ランタイムが急速に成長する。上記のアンダーサンプリング式を調べると、視点がボリュームのフロントに接近するにつれて、分子がゼロに接近することがわかる。リア・スライスに実施される仕事量もゼロに接近する。アンダーサンプリング方法のランタイムは、視点がボリュームにより近くなるにつれて、減少する。

#### 【0183】

視点ジオメトリに関係なく、本発明のER透視アルゴリズムによって実施される仕事量は、1スライス当たり  $O(N^2)$  および  $\omega(4N^2)$  だけ制限される。この方法のいくつかの利点は、アルゴリズムのランタイムについての上限がボクセルの数についてリニアであり、視野位置から独立しており、また、達成される画質についての下限も視野位置とは無関係であるということである。したがって、ユーザは、所望の画質について基本サンプリング・レートを設定することができ、この所望の画質についてのボリュームの全体にわたってサンプリング・レートが充分であることが確実になる。

#### 【0184】

それと対照的に、普通のオーバーサンプリング方法は、画質についての下限を提供するが、アルゴリズムのランタイムは、本発明のER透視方法のランタイムよりもかなり大きくなり得る。普通のアンダーサンプリング方法は、レンダリングのためのランタイムについての上限を提供するが、画質はER透視方法よりもかなり悪くなる可能性がある。

#### 【0185】

図23を再び参照して、ここには、本発明による、好ましい後から前のER透視なレイ・キャスト・アルゴリズムが示してある。図23のアルゴリズムは、擬似コード表現として示され、Z主要軸線投影を採る。本発明のER透視アルゴリズムは、均一で規則的なグリッド上へ透視投影を実施するときに伝統的な落とし穴に苦しむことがない。このユニークな方法は、オーバーサンプリング方法よりも急速に実行でき、アンダーサンプリング方法よりも良好な画質を与える。レイを合併するためのバートレット・フィルタを使用することで、普通ボックス・フィルタよりも画質を向上させることができる。ER透視アルゴリズムは、入

力データについての効果的フィルタリングを特徴づけることによって適格となる。

#### 【0186】

本発明の別の形態によれば、大きなボリュームをレンダリングする方法が提供される。この方法では、ボリューム・データセットが本発明のCube-5装置の物理的なシングル・パス能力を超える。この好ましい方法は、ボリューム・データセットを複数の立方体レンガに細分する。所定のオーダーでレンガを移動させることは、好ましくは、レンダリング前に先のレンガのベースプレーン画像でCube-5装置の合成バッファの初期化を可能にする。それによって、レイ・パスおよび合成は、ボリューム全体を通じてロジック的に拡張される。レンガ間の境界に関する情報は、好ましくは、正しいサンプリングを保証するために再読み込みされる。この方法を使用して、最大ボリューム・サイズは、利用できる中間ベースプレーン記憶量によってのみ制限される。

#### 【0187】

透視投影中に複数のボクセルが同じ画像ピクセルに寄与するデータセットの領域においては、同等の品質の画像が、好ましくは、ディテール・レベル（LOD）ツリーを使用してレンダリングされる。このツリーは、たとえば、事前処理段階で増大する近傍サイズのボクセルを結合することによって生成してもよい。LODを利用して単一の大きいボリュームを透視レンダリングしている間、好ましくは、視点にかなり近い、ボリュームの小さい部分だけが、最高精度で読み込まなければならない。視点に関して、ボリュームのより遠い部分を、次に、データのより低い解像度バージョンからレンダリングされ得る。したがって、フレーム率および／またはデータセット・サイズは、好ましくは、増大する。本発明の透視アルゴリズムにおける各領域（先に述べた）は、今や異なったLODのところにあるので、領域間でレイを濾過する必要はもはやなく、レイを再分布させることだけが必要である。好ましくは、各LODツリー・レベルの1つの領域だけが処理され、したがって、これらの領域だけをメモリ内にページングしなければならない。

#### 【0188】

本発明のディテール・レベル（LOD）方法は、また、視点から異なった距離で複数のオブジェクトからなるシーンをレンダリングするのにも使用できる。このような場合、開始LODは、好ましくは、スクリーン・スペース画像とほぼ同じサイズのベースプレーン画像をデリバリーし、レンダリング時間を画像解像度に関連付け、オブジェクト・サイズに関連付けない（すなわち、スケール独立性）ように選ぶ。

#### 【0189】

後から前へのレンダリングも同様に本発明の意図したものであり、本発明の範囲内に入っているが、ユニークなLOD方法は、ここでは、前から後へのレンダリング・コンテキストとして説明する。前から後にレンダリングすることで、レンダリングしようとしているボリュームの最も詳細な表現のスラブで過渡することが好ましい。本発明の好ましい方法においては、ボリューム・スラブの厚さは、スラブのフロントおよびバックにおける投影ボクセル距離が、先に説明したような本発明の透視投影と同様に、2の因数だけ異なるように選ぶ。スラブをレンダリングした後、現合成バッファ画像をワープ単位で0.5の因数だけスケールアップすると好ましい。これは、半解像度の次のスラブのレンダリングのために合成バッファを初期化する。好ましくは、各LODのスラブは、1つだけ、実際にレンダリング・パイプラインを流れて流れる。したがって、大きいボリュームの場合、これらのスラブだけが、オンボード3Dメモリ内にページングされなければならない。

#### 【0190】

ここで、本発明の装置がオフライン計算（たとえば、ディテール・レベル（LOD）の生成やデータセットのフィルタリング）をスピードアップするのにも使用できることは明らかである。LODを生成するためには、本発明のトリリニア補間ユニット（Trilin）は、好ましくは、そのすべての重みを0.5にセットする。ひとたび新しいサンプルを利用できるようになったならば、これらのサンプルを、好ましくは、サブサンプリングし、新しいボリュームにコンパクト化する。これが、次のより粗いLODとなる。データセットを濾過するために、トリリニア補間ユニットは、再び、0.5の重みだけを使用する。しかしながら、この

とき、データは圧縮なしにレンダリング・パイプラインの始めに戻される。各付加的なパスは、すべての主要軸線方向にもう1つのボクセル範囲を有するフィルタ・カーネルを備える新しい濾過済みのボリュームを創り出す。

#### 【0191】

より高い品質の画像をレンダリングするために、本発明の装置および方法は、好ましくは、所望のトレードオフに依存して、フル・ハードウェア手段、マルチパス・アルゴリズムおよび／またはこれら2つの組み合わせを利用する融通性を提供する。フル・ハードウェア手段およびマルチパス方法は、好ましくは、2つの主要機能領域、すなわち、フィルタリングと補間におけるより正確な計算を提供する。

#### 【0192】

Cube-4アーキテクチャ、すなわち、本発明(Cube-5)の先行モデルは、或る特定の場所での $x$ 、 $y$ 、 $z$ 勾配の各々を評価するために2つだけのサンプル・ポイントを持つ中央差分勾配フィルタを利用する。より大きい3Dフィルタが、より正確な勾配評価をデリバリーすることができる。たとえば、Sobelフィルタ(中心ポイントからのマンハッタン距離の逆数から誘導した重みを持つ $3^3$ フィルタ)である。しかしながら、 $3^3$ フィルタの直接的なハードウェア機能は、27台の乗算器と26台の加算器を必要とする。

#### 【0193】

本発明の装置は、対称形回旋フィルタを使用することによってこの高価な従来技術方法の代替案を提供する。この回旋フィルタは、3台の乗算器と6台の加算器のみを能率的に備えることが可能であり、これはコスト節減にとって意味があることである。1勾配成分当たりのハードウェアの複製は、好ましくは、代わりに3パス・アルゴリズムを適用することによって避けることができる。たとえば、図25は、Sobel勾配フィルタの $x$ 成分の対称形近似を示している。各ステージ内で、重みは、好ましくは、合計前に最も近い隣接体に適用される。図25を参照して、各ステージが生データの代わりに前のステージの出力に作動する場合、図25に示した重みは、Sobel勾配フィルタの $3^3$ 対称形近似(図25の右側)を効果的に生成する。 $x$ 重みを $\{1\ w\ 1\}$ に変えることで、その代わ

りに、ガウス・フィルタの近似を生成することになる。

【0194】

本発明は、付加的なハードウェアを必要としないが、フレーム率を低下させるより高い品質のレンダリング・モードを意図している。このような例の1つは、ディテール・レベル（LOD）情報を利用することによって勾配評価へのより大きい近傍寄与率を達成することである。中央差分勾配が次のより粗いLODのデータについて計算される場合、 $6 \times 4 \times 2$  フィルタを同等に効果的に使用する。この場合、6は、現勾配成分の方向における範囲にある。本発明の装置（すなわち、Cube-5アーキテクチャ）は、データのミップ・マッピングしたLOD表現を保持することができるので、このフィルタは、単純な中央差分を越えて、本質的にハードウェアの増加なしに、達成することができる。

【0195】

本発明によって提供される別のより高い品質のマルチパス・レンダリング・モード（付加的なハードウェアは必要ない）は、医療分野やその他の分野で有利な用途を有するトリキュービック補間の近似である。このモードは、より正確な再サンプリングおよびイソポジション計算を可能にする。これのために、本発明は、好ましくは、ピースワイズ $4^3$  ボクセル・フィルタを一連のリニア補間および外挿に分解する。これは、すべてのディメンジョンで対称的であり、中間結果の効果的な再利用を可能にする。

【0196】

より高い品質のレンダリングを実施する際して、多パス・アルゴリズムを使用するのとは逆に、Cube-5パイプライン内により正確で融通性のある勾配評価を与えるための付加的なハードウェアを使用することには、或る種のトレードオフがあることは了解されたい。一般に、多パス・アルゴリズムを使用することで、本発明のアドレス生成・制御ユニット（図5参照）を変えて計算目的のためのパイプラインを一時的に停止させることが必要であり、それと共に、ハードウェア方法が付加的な特定用途向け集積回路（ASIC）ロジックと、より大きい近傍をサポートするための付加的なコネクションとが必要である。

【0197】

強化したボリューム・レンダリング能力に関して、本発明の好ましい実施例は、任意のプレーンによるクリッピングをサポートする。各プレーンからの距離は、好ましくは、1プレーン当たりレジスタと1つの加算器だけを使用して増分計算することができる。目で見えるように正方向だけを定める普通のクリッピング・プレーンに加えて、本発明の装置は、好ましくは、所定オフセットの外側に位置するすべてのサンプルを無効にすることによって、斜めのマルチ平面再フォーマット化(MPR)のためのデータセットから任意の厚いスライスを抽出することをサポートする。

#### 【0198】

軸線整合したカットティング・プレーンが、当該立方体に対して横方向にボリュームを制限することによって、実施されると好ましい。あるいは、本発明は、ボリュームからの単純な立方体を排除するようにこの横断を制限すること(たとえば、ボリュームのほとんどの八分円を視覚化すること)も意図している。

#### 【0199】

クリッピングに加えて、本発明は、さらに、オブジェクトのカラーを変調して、たとえば、半透明の媒体を通る光の大気減衰をシミュレーションするデプス・キューイングを意図している。当業者によって明らかなように、この現象は、媒体も或る種のカラー(たとえば、白色または灰色)に寄与するとき、霧またはかすみと呼ばれる。本発明に従ってこの特徴を実施するために、通常、転送機能を修正することによって、はっきりした領域を半透明なカラーと共に置くと好ましい(たとえば、デプス・キューイングの場合には黒、霧の場合には白)。各最終ピクセルは、好ましくは、さらに、減衰され、視点からボリュームの表面までの距離を勘案する。これは、ワーピングの一部として実施されると好ましい。

#### 【0200】

本発明の装置は、さらに、x、y方向における2の好ましいデフォルト・スーパーサンプリング・レートでスーパーサンプリングした画像のレンダリングをサポートするが、他のサンプリング・レートも考えられる。画質をさらに向上させるために、各レイに沿ったサンプリング・レートを増大させてもよい。どちらの方法も、3Dメモリからボクセルを再読み込みする必要がない。本発明の装置は



、好ましくは、ボリューム横断オーダーを変え、既にバッファ内に存在するボクセルを繰り返し読み出すようにする。ボクセルを再利用されるたびに、新しい重みが、好ましくは、新しい再サンプリング位置を反映するように本発明のトリリニア補間ユニット (Trilin) で利用される。

#### 【0201】

本発明における好ましいスーパーサンプリング実施例においては、中央差分勾配が、1 距離単位隔たった隣接体間で計算され、十分な精度を保証する。これらの勾配は、好ましくは、最初に差分を取り出し、その後補間することによるか、または、最初に補間し、次いで、 $k$  個の位置隔たった ( $k \times$  オーバーサンプリングと仮定する) 隣接体間の差分、好ましくは中間隣接体のではない差分を採用することによって計算される。新しい  $\alpha'$  値を計算するとき、分類ステージは、新しいサンプル間距離を考慮しなければならない。したがって、スーパーサンプリング中、ボリュームは、好ましくは、各スライス内のインタリーブされたパターンにおいて横断させられることになる。これは、本質的に、半透明の材料 (ゲル) がサンプリング・レートから独立して蓄積された不透明度 ( $R G B \alpha$  値) を保つことを保証する。したがって、たとえば、 $z$  方向における  $k$  のオーバーサンプリング因数の場合、修正済みの  $\alpha'$  値が使用されると好ましい。ここで、 $\alpha' = 1 - (1 - \alpha)^{1/k}$ 。

#### 【0202】

異方性データセットは、異なった軸線に沿ったサンプル間の異なった距離を有する。したがって、勾配計算および最終的な二次元 (2D) 画像ワープは、好ましくは、軸線依存スケーリング因数を必要とする。それに加えて、サイト・レイがボリューム・データセットを通してキャストされつつある方向は、好ましくは、暗黙のボリューム・スケーリングを勘案するように調整を必要とする。これは、等方性グリッドに異方性データを記憶するときに生じる。 $\alpha'$  値は、好ましくは、サイト・レイがボクセル・セルを通して移動する方向依存距離  $d$  に従って、調整される。訂正された  $\alpha'$  は、 $\alpha' = 1 - (1 - \alpha)^d$  である。ここで、方向依存距離  $d$  は、好ましくは、範囲

#### 【0203】

## 【数 4 4】

$$\{1, \sqrt{3}\}$$

内にある。

## 【0204】

上記のボリューム・レンダリング能力を強化する方法に加えて、本発明は、さらに、ポリゴンとボリュームの混合、ポリゴンのボクセル化、複数のオーバーラップしているボリュームのレンダリング、テクスチャ・マッピングの実施および画像ベース・レンダリングの加速を含むユニバーサル三次元（3D）レンダリングを行ういくつかのユニークな方法を提供する。これらの方法を、より詳しく以下に説明する。

## 【0205】

本発明の重要な局面は、単一の画像においてジオメトリック・オブジェクト（すなわち、ポリゴン）およびボリュームを正しく混合するそのユニークな能力にある。本発明の装置（すなわち、Cube-5）は、好ましくは、普通のジオメトリ・ハードウェアに影響を及ぼして、Cube-5ボリューム・レンダリング・パイプラインと共に不透明、半透明のポリゴンをレンダリングする。

## 【0206】

本発明の好ましい方法においては、ボリュームおよび不透明なポリゴンを含むシーンをレンダリングするために、すべての不透明なポリゴンを、まず、Zバッファ上に投影する。このZバッファは、所定のベースプレーンと一致し、ボリューム・サンプル距離と一致するに十分な解像度を有する。Zバッファを使用することで、好ましくは、ボリュームのどのスライスがベースプレーン画像の各ピクセルについてポリゴンのフロントにあるかについて判定がなされる。次に、好ましくは、合成バッファをプリロードする（すなわち、初期化する）。このとき、この投影されたRGBαZ（すなわち、Zバッファ）画像は、ポリゴンのカラーおよびデプス画像を表している。その後、ボリュームを合成バッファにおいて可能にされたz比較でレンダリングする。不透明なポリゴンのデプス値をチェックし

て、不透明なポリゴンで隠されたボリューム・サンプルを最終画像に寄与させ続ける。最終的に、不透明なポリゴンは、背後にボリュームを隠し、フロントにあるボリュームがポリゴン上に正しく合成される。

#### 【0207】

換言すれば、本発明の好ましい方法によれば、合成バッファは  $z$  バッファ画像  $\{C_z, Z_z\}$  でプリロードされる。ここで、 $C_z$  は、ジオメトリ・サンプルの値を表しており、 $Z_z$  は、所定視点からジオメトリ・サンプルのデプスを表している。後から前への合成中、ボリューム・サンプルがジオメトリの後にあるとき（すなわち、サンプル  $Z_s$  のデプスがジオメトリ・デプス  $Z_z$  より大きいとき）、合成バッファ  $C_{out}$  における生じた出力ピクセルは、好ましくは、ジオメトリ・サンプル値  $C_z$  に等しくなる。同様に、前から後への合成中、当業者には明らかなように、サンプルは、好ましくは、Porter-Duff over operatorを使用して合成される。Porter-Duff  $\alpha$  合成ルールについての詳しい説明は、たとえば、テキスト *Compositing Digital Images*, by T. Porter and T. Duff, *Computer Graphics (SIGGRAPH84)*, vol. 18, no. 3, pp. 253-259, July 1984にある。この文献は、参考資料としてここに援用する。したがって、合成バッファの結果生じた出力ピクセル  $C_{out}$  は、好ましくは、ボリューム・サンプルがジオメトリのフロントにあるとき（すなわち、ボリューム・サンプル  $Z_s$  のデプスがジオメトリ・デプス  $Z_z$  より小さいとき）、ジオメトリ・サンプル値  $C_z$  を通じてボリューム・サンプル値  $C_s$  に等しくなる。

#### 【0208】

すべての断片（半透明のポリゴン・ピクセルおよびボリューム・サンプルの両方）は位相的デプス・ソーテッド・オーダーで描かれなければならないので、半透明ポリゴンは、より複雑な状況を採る。これは、over operatorを持つ合成半透明断片は交換可能でないので、必要である。したがって、ポリゴンは、シーンまたは視野ジオメトリが変わるときはいつでも、リデプス・ソートされなければならない。さらに、ソーティングは、デプス・サイクルの取り扱いを含めて、位相的に正しくなければならない。

#### 【0209】

或る種のハードウェア・ソーティング・サポートを与えるためにAバッファを使用するアーキテクチャが提案されているが、ハードウェアにおいてAバッファを実装することで、一回だけのパスでほんの限られたデプス複雑性（すなわち、1ピクセル当たりのオーバーラップしているポリゴンの数）を可能にするが、これは高価である。たとえば、普通のAバッファ・アルゴリズムの説明は、たとえば、テキストThe A-Buffer, an Antialiased Hidden Surface Method, by L. Carpenter, Computer Graphics (SIG GRAPH 84), vol. 18, no. 3, pages 103-108, July 1984に見出すことができる。

#### 【0210】

好ましい方法において、本発明は、スライス・オーダー・レイ・キャストリングにポリゴン・レンダリングを適応させ、ポリゴン毎あるいはピクセル毎よりもむしろ、ボリューム・スライス毎に全レンダリング・プロセスを同期化する。Cube-5装置は、好ましくは、図26に示すように、ボリューム・サンプル212のスライス間にインタリーブされた、あるいは、ぴったり適合する薄いスラブのジオメトリック・オブジェクトをレンダリングするのにジオメトリ・パイプラインおよび普通のグラフィックス・ハードウェアを利用する。

#### 【0211】

次いで図26を参照して、ボリュームの各スライスは、好ましくは、ボリューム保存軸線に対して垂直なプレーンでサンプリングされる。これらのプレーンは、目または視点214から最も遠いところから目に最も近いところまでデプス・オーダーで（たとえば、遠近クリッピング・プレーンを使用することで）描写される。したがって、ボリュームメトリック・データと半透明のポリゴンを混合するために、ポリゴン210の薄いスラブは、好ましくは、ボリューム・サンプル212のスライス間にレンダリングされ、合成される。ここで、スラブ210がボリューム・サンプル・プレーンの2つの連続したスライス間に位置する半透明オブジェクトのすべてを表すことは明らかである。スラブの境界は、好ましくは、すべてのレンダリングされたスラブ210の結合が任意の領域（たとえば、図26に示すような、□, □, □）を失ったり、重複させたりすることがないように定められる。ボリューム・スライスおよび半透明ポリゴン・スラブ210からの

データは、交互に相互にぴったりと適合させられる。こうして、すべての寄与している実体の正しいデプス・オーダリングが保存され、それらを合成するのにover operatorを使用して最終画像ピクセルに正しいカラーを創り出す。

#### 【0212】

本発明の好ましい方法によれば、不透明なポリゴンは、まず、Zバッファリングで描かれる。任意のボリウム・スライスを描く前に、ボリウム範囲の後に位置する半透明のポリゴンが、好ましくは、任意普通の半透明ポリゴン・レンダリング・アルゴリズム（たとえば、ペインタ）を使用して不透明ポリゴン上に描かれる。同様に、ボリウムのフロントに位置する半透明ポリゴンは、アルゴリズムの混合部分の後に描かれると好ましい。ボリウム境界内にデプス方向に位置するが、ボリウムのトップ／ボトム／サイドまでデプス方向に位置するポリゴンは、好ましくは、ボリウム・スライスが半透明ポリゴンを無限にカッティングするように延びるプレーンであったかのように、スライス・オーダーで描かれる。

#### 【0213】

OpenGLを使用して、半透明ポリゴン・オブジェクトの薄いスラブを直接的にレンダリングしてもよい。ポリゴンは、OpenGLに含まれるゲーロー・シェーディング模型を使用してシェーディングされると好ましい。素朴な方法としては、すべてのスラブについて半透明ポリゴンの完全セットをレンダリングし、データの現行の薄いスラブを切るようにより高くてもっと遠くのクリッピング・プレーンをセットすることである。しかしながら、<sup>3</sup>n ボリウムの場合、レンダリングしなければならないn個までの薄いスラブがあり得る。代表的なシーンが薄いスラブのすべてにわたる極めて少ないポリゴンを含んでいるので、本発明は、ポリゴンをスラブ境界にクリッピングし、各スラブ内のポリゴン部分をレンダリングするだけの代替方法も意図している。これは、実質的に、ポリゴン・パイプラインにかかる処理負荷を減らす。しかしながら、各薄いスラブの2つのプレーンに対してすべてのポリゴンをクリッピングするアプリケーションが必要となる。

#### 【0214】

図27に示すように、本発明は、2つのクリッピング・プレーン216、21

8が平行であって、プレーン間に位置するポリゴン部分のみを保持するという事実の利点を採用することを意図している。これは各プレーンに対して別個にクリッピングするよりも少ないポリゴンを創り出すけれども、なお、劇的に三角形カウントを増やすことができる。第1のケースは、三角形220が薄いスラブと交差するが、頂点はスラブ境界216、218内にはまったくないときに生じる。これが生じたとき、1つの頂点は、スラブの片側にあり、他の2つ頂点は、スラブの反対側にあり、2つの三角形に分解される台形を創り出さなければならない。次に、三角形の1つの頂点がスラブ内にあるときを考える。1つの状況において、残りの2の頂点が現スラブの同じ側に位置するように、三角形222はスラブと交差し、たった1つの三角形を作る。第2の状況において、残りの2つの頂点が現スラブの反対側に位置するように、三角形224がスラブと交差する。これは、五角形または3つの三角形を生じさせるので、最悪のケース状況である。最終ケースは、2つの頂点が同じスラブ内に位置するように三角形226がスラブと交差するときに生じ、ここでも再び、台形が創り出され、2つの三角形を生じさせることになる。

#### 【0215】

本発明の好ましい実施例において、半透明ポリゴンにバケット・ソーティング方法が適用される。視野ジオメトリが変わるときはいつでも、ボリューム・サンプルの配置がそれらのジオメトリに対する位置を変える。したがって、本発明は、好ましくは、2つのボリューム・サンプル・プレーン間に各薄いスラブについてバケットを創り出す。シーンにおける半透明ポリゴンのすべては、好ましくは、横断され、各ポリゴンが、その交差するスラブの各々についてバケット内に置かれる。たとえば、図28に示すように、三角形T1は、全部で6つのスラブS1～S6をスパンしているので、全部で6つのバケット内に置かれる。三角形T2は、スラブS2、S3に対応するバケット内に置かれ、残りの三角形についても同様に行われる。図28に示す例の場合、4つの三角形T1～T4をバケットに入れることで、グラフィックス・パイプラインに送られる三角形が12個となる。それと比較して、三角形がスラブ境界にクリッピングされている場合、20個の三角形がグラフィックス・パイプラインに送られることになろう。

## 【0216】

バケット法に代わる案は、走査変換ポリゴンで利用される能動エッジ・リストと同様の能動三角形リストを作ることである。三角形は、それらが交差する第1のスライスところで能動リスト内に置き、三角形がもはや任意のスライスと交差しなくなったときにリストから外すことができる。各三角形が最初にどのスライスに遭遇したかを示すデータ構造を事前計算すると好ましい。この前処理は、本質的に、バケット法と同じであるが、バケット法では、各スライスについて三角形除去をチェックしなくてもよい。

## 【0217】

本発明の方法の1つ利点は、所定フレーム率を維持するために画質をトレードオフする方を選ぶ用途の場合、ボリュームについて描かれるスライス数が減少するにつれて描かれるポリゴンの数が減少することである。ボリューム・スライス数が減少するにつれてインタースライス・サイズが増加するため、これが生じる。達成されるレンダリング率は、描かれるポリゴンの数および描かれるボリューム・サンプルの数（これは、描かれるボリューム・スライス数に比例する）にほぼ比例する。このトレードオフから生じる画質劣化は、任意のボリューム・レンダリング・アルゴリズムにおいてより少しのサンプルを取り出すことと同様に、ボリューム・データにのみ影響を及ぼす。

## 【0218】

半透明のジオメトリおよびボリュームを混合するとき、1つの薄いスラブ内の同じピクセルに描かれつつある2つまたはそれ以上の半透明ポリゴンを取り扱うためオプションが少なくとも3つ存在する。第1のオプションにおいて、ポリゴンは、over operatorをもって規則的な処理オーダーで描くことができる。この方法は不正確なカラーを生成する可能性があるが、ポリゴンが薄いスラブ内にバケットティングすることによってなおソートされるので、カラー・エラー量は制限される。

## 【0219】

2つまたはそれ以上の半透明ポリゴンを取り扱う別の方法は、オンザフライ・ボクセル化として2つのボリューム・サンプル・スライス間に半透明ポリゴンの

薄いスラブを描くことである。普通のボクセル化方法では、表面が3Dボリューム・グリッドに変換された3D走査であるとき、グリッドの解像度は、普通、単一のボクセルのサイズが、レンダリング時に人間の目によって識別され得る最も小さい面積を表すように、選ばれる。X、Yディメンジョンにおいて、ポリゴンは、スクリーン解像度まで描かれる。Zディメンジョンにおいては、各ボリューム・サンプルが十分なスライスをもってレンダリングされ、各ボリューム・サンプルもまた人間の目によって識別され得る最小面積を表すと仮定される。したがって、Zディメンジョンにおける2つのボリューム・スライスによって境される各ピクセルもこの小さい面積を表す。

#### 【0220】

前述のことに鑑みて、本発明の一実施例に従って実施される方法は、3Dグラフィックス・ハードウェアを利用することによってオンザフライ・ボクセル化を計算するものと見ることができる。ボクセル化方法は、2つの好ましい方法のうち的一方を使用することによって、ポリゴンを単一のボクセルに結合する。第1の方法は、各カラー・チャンネルのmaxを取り出すことである。第2の方法は、

#### 【0221】

#### 【数45】

$$C_v = \frac{(C_{p1}D_{p1} + C_{p2}D_{p2})}{(D_{p1} + D_{p2})}$$

として重み付きmaxを取り出すことである。ここで、 $C_{p1}$  は、第1ポリゴン（ポリゴン1）のカラーであり、 $D_{p1}$  は、ポリゴン1の密度であり、 $C_v$  は、ボクセルに割り当てられたカラーである。たとえば、多くのOpenGL手段は、maxをglBlendEquationext (gl\_max\_ext) とブレンドするのを可能にする。密度がアルファ値（たとえば、ボリューム・レンダリングのためのリニア・ランプ転送機能）に等しいと仮定するならば、カラーは、好ましくは、glBlendFunc(gl\_src\_alpha, gl\_one)を用いることによってブレンド前にそれらのアルファ値で重み付けすることができる。しかしながら、OpenGLは、蓄積後にアルファ値の合計によっ



て割ることができないので、完全な先の式を計算することもできない。

#### 【0222】

1つの薄いスラブ内の同じピクセルに2またはそれ以上の半透明ポリゴンを描画する第3の方法は、最も正確な方法と考えることができる。デプス・ソーティング（たとえば、BSPツリー）を実施するために本発明の先に述べた方法の1つを利用することによって、各スラブ内のすべての半透明ポリゴンを適切にオーダリングし続けることができる。デプス・サイクルは、好ましくは、区割りで使用されるプレーンをスパンするポリゴンを分割することによってBSPアルゴリズムで取り扱われる。そして、結局、サイクル内のポリゴンの1つが区割りプレーンとして使用される。

#### 【0223】

前述のように、本発明のCube-5の重要な特徴は、少なくとも1つのジオメトリ・パイプラインまたはエンジンをCube-5システムに接続するユニークな能力である。本発明によれば、PCクラス・マシンについて請求項記載のCube-5システムに1つまたはそれ以上のジオメトリ・パイプラインを接続する2つの好ましい方法が、以下に説明するように、提供される。両方法は、不透明および／または半透明ポリゴンをボリュメトリック・データと混合するユニークな方法である。

#### 【0224】

ここで、不透明なポリゴンが、好ましくは、ボリューム・データセットを通して投影した後、ワーピングが最終画像上に正しいフットプリントを創り出すということは明らかであろう。さらに、Zデプス値は、好ましくは、処理軸線に沿って整合させられ、その結果、ボリューム・スライス・インデックスを・チェックのために使用できる。

#### 【0225】

本発明の一実施例によれば、好ましい方法は、現視野方向について主要視野軸線を決定することによって開始する。図29に示すように、主要視野軸線230が、たとえば、Z軸に沿うように、好ましくは、変換がジオメトリ228に適用される。次に、視点またはアイ・ポイント232は、好ましくは、X軸まわりの

所定角度 $\alpha$ とY軸まわりの角度 $\beta$ の分だけ、注視ポイント234とアイ・ポイント232との間でベクトルを回転させることによってこの方向に沿うように移動させられる。好ましくは、 $\alpha$ 、 $\beta$ は、常に、 $-45 \sim +45$ 度の範囲にある。そうしないと、異なったベースプレーンが選ばれることになろう。XおよびYに沿ったZスライス・シャー変換（「Xに従うX、Y」シャーとしても知られている）は、好ましくは、その後に、次のように視野マトリックスに適用される。

【0226】

【数46】

$$\begin{bmatrix} 1 & 0 & \tan \alpha & 0 \\ 0 & 1 & \tan \beta & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

このジオメトリについて、不透明なポリゴンを描くとき、ポリゴン・フットプリントが「プレワープされる」。その結果、Cube-5レンダリングの終りでワーピング作業が最終的な画像に正しいポリゴンを創り出す。さらに、計算されるZデプスは、好ましくは、処理軸線に沿って距離に比例する。（たとえば、すべての不透明ジオメトリがボリューム範囲内に適合するならば）、ボリュームのエッジにあちこちのクリッピング・プレーンをセットすることが可能となり、そして、デプス・バッファの精度が同じである場合、計算されたデプスは、デプス・チェックについて正確にボリューム・スライス・インデックスとなる。さもないければ、計算されたデプスがボリューム・レンダリング・システムによって利用されるとき、単純なスケーリングを適用しなければならない。しかしながら、この方法を使用するとき、シャーリングは光を正しい場所に移動させることができないので、光位置を考慮しなければならない。

【0227】

半透明ポリゴンの薄いスライスは、好ましくは、スペース内のそれらの3D位置とジオメトリック的に整合する。好ましくは、アイ・ポイントが、前述したよ

うに、最初に整合させられる。次に、オブジェクトが最終画像プレーンへ全面的に投影するのを防ぐために、ジオメトリは、好ましくは、現在の薄いスラブの中心がシャーリング前に $Z=0$ となるように並進させられる。クリッピング・プレーンによって、現在の薄いスラブだけをレンダリングすることができ、投影プレーンは、たとえば、`glOrtho`（透視の場合、`glFrustum`）を持つ領域を境する2つのボリューム・スライス内にあるようにセットされる。

#### 【0228】

本発明を理解するには、フレーム・バッファ設計およびコンポスティング・バッファ設計の構成を理解することが重要である。図30に示すように、本発明のCube-5ボリューム・レンダリング・パイプライン236は、好ましくは、コンポスティング・バッファと呼ばれる緊密に接続されたオンチップSDRAMバッファ238を利用し、ボリュームがスライス・オーダーで処理されるときに部分的に合成されたレイを保持する。このアーキテクチャは、スライス・オーダー・レンダリングに固有の規則的な処理シーケンスを利用する。具体的には、ボリューム240の各スライスは、好ましくは、各行の先の最も左のボクセルから最も右のボクセル、そして、各スライスの最も下の行から最も上の行一番上の列（おそらく若干のスキューを持つ）と同じオーダーで処理される。こうして、SDRAMコンポスティング・バッファ238は、スライスのサイズに等しい長さを有する単純なFIFO待ち行列になる。SDRAM待ち行列は、好ましくは、32ビット長であり、8ビットの固定ポイントRGB $\alpha$ 値（コクセルと呼ばれる）を保持することができる。各パイプライン236は、好ましくは、待ち行列のフロントからコクセルを読み込み、1つのコクセルをクロック・サイクル毎に待ち行列のリアに書き込む。

#### 【0229】

それと対照的に、次に図31を参照して、普通のPCクラス・ジオメトリ・パイプライン242は外部のDRAMフレーム・バッファ244を利用する。これは、各ピクセルについてRGB $\alpha$ カラー値およびZデプス値を記憶する。ポリゴン・レンダリングがスライス・オーダー・ボリューム・レンダリングに固有の規則的なアクセス・オーダーリングを好まないのも、このフレーム・バッファ24

4は、ランダム・アクセスをサポートしなければならない。ノーマル・ポリゴンレンダリングは、10～50個のピクセル間で平均のスクリーン上に三角形を生成する。したがって、DRAMメモリは、このサイズのスクリーンの面積へのアクセスを最大にするように構成される。

#### 【0230】

図31に示すように、本発明の3Dテクスチャ・マッピング方法をジオメトリ・パイプライン242に実施したとき、スクリーンに対して垂直なボリューム・スライス246は、ボリュームを通してテクスチャ・マッピングされる。ボリューム・スライス246についてのペル頂点ジオメトリ計算は、任意レベルのグラフィックス・ハードウェアで容易に達成することができる。しかしながら、テクスチャ・メモリ248およびフレーム・バッファ244の両方に対するランダム・アクセスをサポートする要件は、DRAMフレーム・バッファで達成できる塗り潰し率にこの方法の性能を限定する。

#### 【0231】

非常に高い端面グラフィックス・システムは、代表的には、ポリゴン・パイプラインの断片処理セクション250における大規模な平行度を利用する。これは、非常に分散されたフレーム・バッファと相まって、塗り潰し率性能を向上させることができる。

#### 【0232】

図32には、本発明に従って、Cube-5ボリューム・レンダリング・システム252にジオメトリ・パイプライン242を接続するための一実施例が示してある。図32に示すように、SDRAMコンポスティング・バッファは、好ましくは、Cube-5パイプライン252内部から除去され、その代わりに、外部のDRAMフレーム・バッファ254を使用する。普通のポリゴン・エンジンと同様にDRAMフレーム・バッファ254を構成するというよりもむしろ、本発明のフレーム・バッファにおけるメモリは、好ましくは、それが特にボリューム・レンダリングについて最適化されるように構成される。フレーム・バッファ254は、また、好ましくは、3Dグラフィックス・パイプライン242からアクセスでき、ボリュームとのポリゴン・データ256の混合を可能にする。

## 【0233】

図32を続いて参照して、二重用途フレーム・バッファ254は、好ましくは、2つのパイプライン242、252を接続する。好ましい方法において、不透明、半透明ポリゴンの両方およびボリューム・データを持つシーンをレンダリングするために、ジオメトリ・パイプライン242は、まず、Zデプスですべての不透明なポリゴンをレンダリングする。ボリューム・メモリ258に記憶されたボリューム・スライスおよび半透明ポリゴンの薄いスラブを、次に、交互に（たとえば、ドブテイル式に）レンダリングする。すなわち、ボリューム・スライスをCube-5パイプライン252で、半透明ポリゴンをグラフィックス・パイプライン242でレンダリングする（不透明ポリゴンも同じドブテイル・アルゴリズムでレンダリングできるが、グラフィック・パイプラインに対する需要は増大する）。

## 【0234】

Zデプス・チェックは、好ましくは、隠れたオブジェクトを正しく除去するのに利用され、ブレンディングが両パイプラインにセットされ、サンプルおよび断片を正しく合成する。ジオメトリ・エンジン242は、好ましくは、本発明のCube-5システムの要求する最終的なベースプレーン・ワープを実施する。

## 【0235】

DRAMバッファ254の設計は、たとえば、 $256^3$ 個のボリューム・データセットの30Hzレンダリングのために必要な毎秒503,000,000個のサンプルを得るのに重要である。したがって、まず、グラフィックス・パイプラインにレンダリング・パイプラインを接続することを説明する前に、Cube-5レンダリング・パイプラインそれ自体に対してDRAMバッファを創り出すと役に立つ。本発明のボリューム・レンダリング・システムは、好ましくは、複数のCube-5パイプラインからなる。各レンダリング・パイプラインにおいて、クロック・サイクル毎に、コクセル（RGBαからなるコンポスティング・バッファ要素）を、SDRAM合成バッファFIFOから読み込み、適切なコンポスティング方程式に従ってブレンドされる。次に、新しいコクセルは、FIFOのリアに置かれる。好ましい実施例においては、コクセルの構造は、32ビット

トのカラー、8ビットの各RGBα、32ビットのZデプス情報、24+8ビットのステンシルを含むように変更される。この構成は、コンポスティング・ステージにおいてZデプス・チェックを取り扱うのに必要である。任意のボリューム・レンダリングが開始する前に、不透明ポリゴン・レンダリングが完了したと仮定すると、32ビットのZデプス/ステンシル情報を読み込むが、書き直しはしない。したがって、あらゆるクロック・サイクルについて、各Cube-5パイプラインは、8バイトのkokセル・データを読み込み、4バイトを書き戻す必要がある。

#### 【0236】

好ましくは、本発明のレンダリング・パイプラインは、16ビットのワード・サイズを持つメモリ・チップを利用する。この構成を使用することで、4つのワードが、サイクル毎に各パイプラインによって読み出されなければならない、2つのワードを書き込まなければならない。これを行うためには、1パイプライン当たり6つの16ビット・メモリ・インタフェースを必要とする。同期DRAM（SDRAM）チップにおける新生の技術（本発明はそれ自体を利用することができる）が、二重データ率（DDR）として知られており、これは、クロックの立ち上がり、立ち下りの両方でデータを読み書きする。DDR SDRAMを使用することで、本発明は、1クロック当たり64ビットのデータを読み込むための2つの16ビット・メモリ・インタフェースを、1クロック当たり32ビットを書き込むために、1パイプライン当たり合計3つの16ビット・メモリ・インタフェースのために1つの16ビット・メモリ・インタフェースを利用できる。

#### 【0237】

次に図33を参照して、クロック・サイクル毎に読み書きが実施されてパイプラインをフルに保持するので、本発明は、好ましくは、1セットのフレーム・バッファ・チップの1セット（たとえば、セットA）から読み出し、別のセット（たとえば、セットB）262に書き込む。Cube-5システムは、ボリュームの完全スライスについてセットA260から読み出し、セットB262に書き込み、次のスライスについてスワッピングすることをいとしている。しかしながらこの方法では、各フレーム・バッファ・チップ・セットは、完全なフレーム・バ

ッファを保持するのに充分に大きくなければならない。さらに、ポリゴン・エンジンは、どのセットが現セットであるかについて指示されなければならない。したがって、好ましい実施例においては、本発明は、1つのフレーム内でセットA 260とセットB 262の間で読み書きを交互に行い、読み込みセットが書き込みセットとなるまで読み込みセットからの処理済みコクセルをバッファする。することを交替させる260、そして、リードからの被処理コクセルがそれまでセットしたフレームおよびバッファ内にセットB 262はライトセットになる。あらゆるメモリ・アクセスはバーストでなければならないので、各バーストは、実際には、4つのクロック・サイクルを持続させ、16ビットDDR DRAMチップで4つのコクセル（すなわち8個のワード）を読み書きする。Cube-5システムは、好ましくは、全部で4つのバンクを通して循環し、新しいRGB $\alpha$ 値を書き込む前にメモリ・バンド幅を飽和状態に保持する。この理由のために、好ましくは、16コクセルFIFO待ち行列264（4つのバンクの各々について4つのコクセル）に、コクセルの新しく合成されたRGB $\alpha$ 部分が記憶される。

#### 【0238】

本発明のCube-5ボリューム・レンダリング・システムにおけるパイプライン等の数については多くの異なった構成の可能性がある。全部で12のメモリ・インタフェースを創り出している4つの平行パイプラインの例を図33を参照しながら以下に説明する。図33に示すように、各パイプラインは、フレーム・バッファのZデプス/ステンシル部分268に対する1つの読み出しインタフェース266と、フレーム・バッファのRGB $\alpha$ 部分のセットA 260、セットB 262それぞれをセットするための2つの読み込み/書き込みインタフェース270、272とを含む。30Hzで256<sup>3</sup>ボリュームをレンダリングするために、4つのパイプラインは、毎秒125,000,000個のボクセルを処理する。したがって、133MHzクロックが、チップおよびSDRAMについて利用される。メモリ・チップ上へフレーム・バッファ・ピクセルをマッピングすることは、性能にとって重要である。それは、Cube-5パイプラインの処理オーダーおよび実質的に同時の4つのパイプラインによる並列アクセスに正確に整合していなければならない。説明をより容易にするために、ここで、Cube-5ア

ーキテクチャのスキュード・メモリ・アクセスが「非スキュー」にされ、ボリューム・サンプルが4つのグループにおける各スキャンラインを横切って左から右のオーダーにあるようにすると仮定する。この設計はスキュード・メモリに拡張できるが、ジオメトリ・パイプラインおよびスクリーン・リフレッシュ・システムは付加的なスキューイングに気づかなければならない。

### 【0239】

図34は、フレーム・バッファのコンセルのRGB $\alpha$ 部分の好ましいレイアウトを示している。所与のスキャンライン274について、セットB278にあるピクセル・グループの前にセットAに存在するピクセル・グループがある。これはスキャンライン274全体を横切って繰り返される。各セットの長さは、各セットが各チップ内部の4つの異なったバンクから読み出されるピクセルを含まなければならないという事実により、64ピクセル分である。そして、各バンクは、4つの平行チップ/パイプラインから読み出された4つのRGB $\alpha$ 値からなる。したがって、フレーム・バッファ内のピクセル・データは、8つのチップを横切ってインタリーブされる。より詳しく言えば、それは、ほんの4つのチップを横切って実際にインタリーブされる。このことは、毎秒、データの

$$4 \text{ パイプライン} \times (1 \text{ RGB}\alpha \text{ チップ} + 1 \text{ デプス・チップ}) \times 16 \text{ ビット}$$

$$\times 133 \text{ MHz} \times 2 \text{ データ・レート} = 34 \text{ Gbits} = 4.15 \text{ Gbytes}$$

を読み出すインタフェースを提供する。これは、必要な

$$256^3 \times 30 \text{ Hz} \times 8 \text{ bytes} = \text{毎秒 } 3.75 \text{ Gbytes}$$

にまざる。ここで、8バイトは、4バイトのRGB $\alpha$  + 4バイトZデプス/ステンスルとして構成される。さらに、フレーム・バッファ・サブシステムは、

$$4 \text{ パイプライン} \times 1 \text{ RGB}\alpha \text{ チップ} \times 16 \text{ ビット} \times 133 \text{ MHz}$$

$$\times 2 \text{ データ率} = 17 \text{ Gbits} = 2.1 \text{ Gbytes}$$

を書くことができる。

### 【0240】

ここで再び、

$$256^3 \times 30 \text{ Hz} \times 4 \text{ bytes} = \text{毎秒 } 1.8 \text{ Gbytes}$$

を取り扱う。これは256<sup>3</sup> ボリュームのリアルタイム30Hzレンダリングに



とって必要である。

#### 【0241】

この特別なバンド幅は、シッティング・アイドルではない。スクリーンは、フレーム・バッファにおけるデータからリフレッシュされなければならない。ここで、60Hzリフレッシュ率での1280×1024スクリーン解像度を仮定し、全部で4つRGBαバイトがフレーム・バッファから読み出される（本発明のバースト・モード・アクセスがどのようにでもそれらを検索するので）と仮定すると、

$$1280 \times 1024 \times 60\text{Hz} \times 4\text{bytes} = 300\text{Mbytes}$$

が毎秒フレーム・バッファから読み出される。フレーム・バッファのRGBα部分だけがリフレッシュについて必要である。したがって、リフレッシュ・データは、8つのチップから読み出される。各チップについて10回のデータ・バースト読み出し／書き込み（セットAかセットBかに依存する）を行い、その後、リフレッシュのために1回データの読み出しを行えば充分である。メモリ・アクセスのこの分布により、データの一貫したストリーム（burstyではあるが）をリフレッシュ・ハードウェアに設けることができる。Cube-5パイプラインは、また、小さいパーセンテージの余分なサイクルを含み、したがって、メモリ・サブシステムが一時的にリフレッシュのために停止させられたときに、30Hz<sup>2</sup>56<sup>3</sup>レンダリングを達成する能力を失うことはない。

#### 【0242】

以下、本発明の好ましい実施例に従って、Cube-5ボリューム・レンダリング・パイプラインにグラフィックス・パイプラインを接続する代替方法を説明する。この好ましいコネクション方法は、グラフィックス・パイプラインおよびボリューム・レンダリング・パイプラインの両方を常時作動状態に維持し、Cube-5チップ内のSDRAM合成バッファのデータを合併する。任意所与の時点で、ボリューム・レンダリング・パイプラインは、コンポスティング・バッファを通して先の薄いポリゴン・データ・スラブで現ボリューム・スライスをコンポスティングし、グラフィックス・パイプラインが、次の半透明ポリゴンの薄いスラブをレンダリングする。

## 【0243】

ここに記載される方法は、なお、上記のように、ボリューム・スライスおよび半透明ポリゴン・データの薄いスラブをドブテイルするユニークな方法を利用する。第1段階において、すべての不透明ポリゴンをベースプレーン（たとえば、スクリーンに対して最も平行なボリューム面）と一致するZバッファ上へ投影する。次に、投影したRGB $\alpha$ Z画像をボリューム・レンダリング・パイプラインのコンポスティング・バッファにロードする。その後、ボリュームをコンポスティング・ステージにおいて可能にされたZ比較でレンダリングする。半透明ポリゴンの薄いスラブは、好ましくは、ジオメトリ・パイプラインによってレンダリングされ、そして、それらの対応するRGB $\alpha$ データは、本発明のボリューム・パイプラインに送られ、ボリューム・パイプライン内でSDRAMコンポスティング・バッファにブレンドされる。

## 【0244】

好ましくは、ボリューム・レンダリング・アクセラレータのコンポスティング・ステージは、1段階当たり2つの層（1つのボリュームと1つの半透明ポリゴン）を合成し、したがって、ボリューム・レンダリング・プロセスを遅延させないように改造される。これは、或る種の余分なロジックの追加を必要とする。旧コクセルcを通して半透明ピクセル断片p上にボリューム・サンプルvを二重合成するための直接的な方程式は、5つのステージにおいて4つの加算と4つの乗算を必要としよう。

## 【0245】

## 【数47】

$$C_c = C_v \alpha_v + [C_p \alpha_p + C_c (1 - \alpha_p)] (1 - \alpha_v)$$

しかしながら、単純な数学を使用することによって、二重合成を、以下の式（いくつかの計算は再使用する）で、6つのステージにおいて4つの加算と2つの乗算を用いて計算することができる。

## 【0246】

## 【数48】

$$C_r = (C_r + (C_p - C_r)\alpha_p) + [C_r - (C_r + (C_p - C_r)\alpha_p)]\alpha_r$$

当業者には明らかなように、ハードウェア・デザイナーは、所与の手段にとって望ましいオプション（すなわち、より少ないロジックとより多いステージ、あるいは、より少ないステージとより多いロジック）を選ぶことになる。

## 【0247】

256<sup>3</sup>のボリュームについてのデータ転送量を考える。好ましくは、ボリュームのフロントに255個のスラブ+1つのバッファがあり、ボリュームの背後に1つのバッファがある。これら257個のスラブは、256KBのデータ（RGB $\alpha$ の256<sup>2</sup>ピクセル）を含む。これは、フレーム・バッファから読み込まれ、フレーム毎に2つのサブシステム間で転送されている64MBと同等である。30Hzフレーム率を達成することは、毎秒1.9GBのバンド幅を必要とする。この多くのデータが十分に広いチャネルで転送され得るが、それは、また、フレーム・バッファから読み出されなければならない。現DRAMフレーム・バッファの構成を変えることなくこの多いデータを読み出すことは実質的に不可能である。さらに、フレーム・バッファは、1フレーム当たり257回をクリアしなければならない。

## 【0248】

このバンド幅問題を解決するために、本発明は、好ましくは、ブランク・ピクセルの実行長コード化（RLE）を使用する。この方法では、各スキャンラインは、個別にコード化される。そして、「run-of-zeros」が、実行長に続く4つのゼロ（RGB $\alpha$ ）としてコード化される。代表的には、シーンにおけるほんの小さいパーセンテージのポリゴンだけが半透明であるから、半透明ポリゴン・スラブは比較的まばらである。これらの薄いスラブにおけるブランク・ピクセルだけ実行長コード化することで、必要なバンド幅において99%以上の低減を行える。好ましくは、本発明の方法は、バンド幅に節減するためにまばらな半透明ポリゴンの2D画像上のRLEを利用する。

## 【0249】

この好ましい方法を使用するで、本発明のCube-5システムにハードウェアを加えることが必要となる。具体的には、付加的なハードウェアは、RLE入力ストリームを復号し、RGB $\alpha$ 断片を創り出すことができるボリューム・レンダリング・パイプラインに含まれ得る。しかしながら、これらの断片が規則的なオーダーでボリューム・パイプラインによって利用されるので、2つのパイプラインを同期化するのに二重バッファを使用して入力ストリームを復号すると好ましい。クロック・サイクル毎に、値は、復号ハードウェアから出力される。ボリューム・レンダリング・マシンが複数のパイプラインを有する場合（現設計の大部分がこれである）、復号ハードウェアは、好ましくは、ピクセル需要を維持することができるように、各パイプラインについて複製される。

## 【0250】

同様に、ジオメトリ・パイプラインに接続される発祥端部でのRLEハードウェアは、リアルタイムでデータをコード化してから、ボリューム・パイプラインに送る。しかしながら、すべての半透明ポリゴンの薄いスラブを読み出し、1フレーム当たり257回フレーム・バッファをクリアするのに、フレーム・バッファへの毎秒1.9GBのアクセスは、なお、必要である。したがって、別々のフレーム・バッファを使用して、データをRLEフォーマットで直接記憶すると好ましい。半透明データの薄いスラブが非常にまばらであるので、ラスタライジングで費やされるよりも多くの時間が、クリアリング、読み出しで費やされる。一般的にラスタライゼーションにとって最適ではないが、RLEバッファは、データのクリアリング、読み出しの両方にとって最適である。たとえば、RLEフレーム・バッファをクリアするには、256<sup>2</sup>フレーム・バッファ全体に書き込みする代わりに、スキャンライン毎に単一のrun of zeros（5バイト）を記憶するだけでよい。

## 【0251】

現ジオメトリ・パイプラインへのインパクトを最小限にするために、RLEフレーム・バッファは、好ましくは、DRAMを埋め込み、それをノーマル・フレーム・バッファに並列で接続する新生技術を用いて実装される。これは、普通、

データが物理的なオーダーで与えられると仮定する普通のコード化・アルゴリズムとは異なる。しかしながら、三角形ラスタライゼーションは、断片の任意のオーダーリングを保証しない。したがって、本発明の装置は、RGB $\alpha$ 値をデータのRLEスキャンラインにランダムに挿入できなければならない。

#### 【0252】

図35は、本発明に従って形成されたRLEインサートのダイアグラムを示している。各断片について、コード化されたスキャンラインは、バッファ毎にコピーされ、新しいRGB $\alpha$ 値を挿入する。ロック・サイクル毎に、単一のフリット（すなわち、RGB $\alpha$ ピクセルまたはrun-of-zeros）が処理される。スキャンライン全体が、フリット毎に処理されると好ましい。図35を参照して、入力バッファ（「in Buffer」）280が、現コード化スキャンラインを保持し、出力バッファ（「out Buffer」）282が、挿入した新しいRGB $\alpha$ 断片を持つ新しくコード化したスキャンラインを保持する。各サイクルで挿入すべきものの選択は、好ましくは、5バイトのマルチプレクサ284によって実施される。本発明の装置は、好ましくは、ポインタ、すなわち、「inPtr」286および「outPtr」288を包含する。これらのポインタは、それぞれ、in Buffer 280、out Buffer 282の両方の現フリットを指摘する。図35の右側にあるロジックは、どのくらい処理されたか（「Total」）290と制御点の2つ（「ctrl\_1」および「ctrl\_3」）を計算する。他のmux制御点（「ctrl\_2」）は、RGB $\alpha$ 値（run-of-zerosについてのフラグ）のすべてを一緒に論理和演算することによって算出される。「xPos」は、断片のx位置として定義される。好ましくは、現バッファがy値毎にメモリ内に位置する場所についてのルックアップ表が実装される。したがって、新しいピクセルを挿入しながらバッファを移動させることができ、表を簡単に更新することができる。この好ましい方法が、図36のRLE\_Add Fragment擬似コード・ルーチンに示してある。図36を続けて参照して、RLE\_AddPixelToScanline機能は、図35に示す本発明のハードウェア実施例で生じる処理を示している。

#### 【0253】

埋め込み式DRAMを利用することによって、本発明は、処理がメモリ・チッ

プで生じるときに利用できる極めて高いバンド幅を利用する。この処理は、DRAM製造プロセスで実施されるのに十分に単純である。たとえば、 $1280 \times 1024$ のフレーム・バッファの場合、必要な最大メモリ量は、50Mbitである。これは、コード化ハードウェアについて3,000,000以上のゲートの余裕をもって、eDRAMダイスに適合する。

#### 【0254】

図37は、ポリゴン・パイプライン242およびボリューム・パイプライン252を、RLEフレーム・バッファ292を通していかに接続するかを示す好ましいブロック図である。RLEフレーム・バッファ292は、好ましくは、二重バッファされ、データの伝送中にレンダリングをすることを可能にある。補助フレーム・バッファが、好ましくは、断片を単に複製することによって既存のものと同一場所で接続され、したがって、ジオメトリ・パイプライン242の残りの部分に影響を及ぼすことがない。ボリューム・パイプライン252は、また、好ましくは、二重バッファリングを利用して先のスラブをブレンドしながらデータの受信を可能にする。ここで、本発明のシステムを用いることで、ボリューム・レンダリングがポリゴンレンダリングと干渉しないことは明らかである。ボリューム・パイプライン252が常に反復可能なオーダー様式でメモリにアクセスするので、それが30Hzボリューム・レンダリングを達成するのに十分な率でフレーム・バッファ内へのサンプル塗り潰し率を達成する。本発明のシステムは、グラフィックス・パイプライン242を利用してボリューム・メモリ258内に記憶されたボリュームをレンダリングする前に不透明ポリゴンをレンダリングする。これは、通常、先のフレームについてボリュームのレンダリングと同時に達成することができる。ポリゴン・エンジンがボリュームと混合した半透明ポリゴンをレンダリングしなければならない場合であっても、通常、ノーマル・シーンにおける少数の半透明ポリゴンによりボリュームが仕上がる前に不透明ポリゴンをレンダリングするに十分な時間がある。

#### 【0255】

本発明の好ましい実施例によれば、プレフィルタリングでボリュメトリック・データセットに三角形を増分ボクセル化し、それによって、正確な多価値ボクセ

ル化を生成する方法が提供される。多価値ボクセル化により、相互混合したジオメトリ、正確なマルチ解像度表現および効率的なアンチエイリアシングで直接的にボリューム・レンダリングを行うことを可能にする。従来のボクセル化方法は、いずれも、バイナリ・ボクセル化だけを計算していたか、あるいは、多価値ボクセル化を非効率的に計算していた。本発明によれば、この方法は、好ましくは、各ボクセル値についてどのフィルタ機能を計算すべきかについて迅速に決定する増分式を生成する。この好ましい方法は、以下により詳しく説明するが、三角形バウンディング・ボックスの1ボクセル当たり8回の加算を必要とする。

#### 【0256】

画像エイリアシングを避けるために、本発明は、好ましくは、スカラー値ボクセルを使用してボクセルの空間占有パーセンテージを表すプレフィルタリングを使用する。二次元ライン・アンチエイリアシング方法の拡張型が、従来公知である (Filtering Edges for Grayscale Displays, by S. Gupta and R. F. Sproull, Computer Graphics (SIGGRAPH81), vol. 15, no. 3, pp. 1-5, Aug. 1981)。中央差分勾配評価のための最適なボリューム・サンプリング・フィルタが表面に対して垂直な一次元向きボックス・フィルタであることも示されている。本発明の方法は、好ましくは、三角形からの距離の単純な一次関数であるこのフィルタを利用する。

#### 【0257】

普通のグラフィック・ハードウェアは、これらの基本的なプリミティブの組み合わせとして表されるより高いオーダーのプリミティブで、ポイント、ラインおよび三角形をラスタライズするだけである。したがって、他のすべてのプリミティブが三角形によって表され得るために、三角形のみをボクセル化することが好ましい。ポリゴン・メッシュ、スプライン曲面、球、シリンダその他をボクセル化のために三角形に細分することができる。ポイントおよびラインは、三角形の特殊なケースであり、本発明のアルゴリズムによって同様にボクセル化することができる。中実オブジェクトをボクセル化するためには、オブジェクトの境界を一組の三角形としてボクセル化するのが好ましい。次に、オブジェクトの内部を、ボリュームメトリック・ファイリング手順を使用して塗りつぶす。

## 【0258】

当業者にとって明らかなように、エッジ関数は、効果的な増分算数によって、エッジからの距離を維持するリニア表示である。本発明の方法は、この概念を3次元へ敷衍し、ボリュメトリック三角形の走査変換中にアンチエイリアシングを適用する。

## 【0259】

本質的に、本発明の三角形ボクセル化方法の一般的なアイデアは、ラスタ・オーダーで三角形のバウンディング・ボックスを走査することによって三角形をボクセル化することにある。バウンディング・ボックス内の各ボクセルについて、フィルタ方程式が、好ましくは、評価され、その結果がメモリに記憶される。式の値は、三角形からの距離の一次関数である。その結果は、好ましくは、ファジー代数ユニオン・オペレータ（すなわち、maxオペレータ）を使用して、記憶される。

## 【0260】

次に図38を参照して、ここには、中実プリミティブ296の中心からライン294に沿った方向付けボックス・フィルタの密度分布が示してある。フィルタの幅は、Wとして定義される。ファジー・セット内の1つのボクセルの包含物は、ゼロと1の間で変化する。方向付けたボックス・フィルタの値によって決定されることを含む。プリミティブ296の表面298は、0.5の密度アイソ・サーフェイス上に位置すると仮定する。したがって、図38におけるように中実プリミティブ296をボクセル化するとき、密度分布はプリミティブ内部の1からプリミティブ外部のゼロまで変化し、エッジのところで滑らかに変化する。表面プリミティブについて、たとえば、図39に示す三角形300について、密度は、好ましくは、表面で1であり、表面から距離Wのところでゼロまで線形に低下する。本発明は、中実体のボクセル化を意図しているが、表面のボクセル化を以下に説明する。

## 【0261】

図39を続いて参照して、フィルタ幅Wについての最適値が、



### $2\sqrt{3}$ ボクセル

単位であることは決まっている（たとえば、Object Voxelization by Filtering, by M. Sramek and A. Kaufman, 1998 Volume Visualization Symposium, pp. 111-118, IEEE, Oct. 1998参照）。シェーディングのために、垂線が、0.5のアイソ・サーフェイスで中央差分勾配を計算することによって推定されると好ましい。中央差分フィルタの全幅がせいぜい

### $2\sqrt{3}$

単位であるため、正しい勾配は、0.5の密度アイソ・サーフェイス上に見出される。三角形300の厚さは、Tとして定義され得る。通常、Tは、厚い表面を望まない限り、ゼロである可能性がある。0.5の密度での閾値決定によって、6・トンネル・フリー・セットのボクセルが、 $W=1$ のときに、生成される。この特性は、ボリュメトリック・フィリング（たとえば、中実オブジェクトを生成するオーダーで）について有用である。

#### 【0262】

1つの三角形について非ゼロ値を持つすべてのボクセルは、好ましくは、バウンディング・ボックス内にあり、このバウンディング・ボックスは、緊密なバウンディング・ボックスよりも、すべての方向において大きい $S=W+T/2$ ボクセル単位である。したがって、本方法の第1段階は、好ましくは、三角形300についての緊密なバウンドを決定し、次いで、それをすべての方向においてSボクセル単位だけ膨らませ、最も近いボクセルまで外側に丸める。

#### 【0263】

図40に示すように、頂点 $C_1$ 、 $C_2$ 、 $C_3$ によって定義される三角形を取り囲んでいる領域は、別々に処理しなければならない7つの領域（たとえば、R1～R7）に分割することができる。本発明の好ましい方法においては、各対象ボクセルは、7つの領域内の包含物についてテストされ、次いで、各領域について異なった式で濾過される。三角形の内部領域R1において、方向付けられたボックス・フィルタの値は、単に三角形のプレーンからの距離に比例している。三角形

のエッジに沿った領域R 2、R 3、R 4において、フィルタの値は、好ましくは、三角形のエッジからの距離に比例する。三角形の角隅R 5、R 6、R 7の領域においては、フィルタの値は、好ましくは、三角形の角隅からの距離に比例する。

#### 【0264】

図40を続けて参照して、領域R 1～R 7は、好ましくは、7つのプレーンからの距離によって区別される。第1のプレーンaは、好ましくは、三角形およびページから外側のノーマル・ベクトルaポイントと同一平面にある。次の3つのプレーンb、c、dは、好ましくは、それぞれ、ノーマル・ベクトルb、c、dを有し、それぞれ、三角形の角隅頂点C<sub>1</sub>、C<sub>2</sub>、C<sub>3</sub>を通過する。最後の3つのプレーンe、f、gは、好ましくは、三角形に対して垂直であり、エッジに対して平行である。すなわち、それらのそれぞれのノーマル・ベクトルe、f、gは、三角形のプレーン内に位置し、内向きとなっており、3つすべてのプレーンからの正の距離が領域R 1を定めるようになっている。プレーン係数のすべては、標準化され、その結果、ノーマルの長さが1となるが、ただし、ノーマル・ベクトルb、c、dについてはそうではなくて、これらは、長さがそれぞれのエッジ長の逆数に等しくなるように標準化される。こうして、プレーンからの計算された距離は、エッジの有効長さに沿ってゼロから1まで変化する。

#### 【0265】

任意の平らな表面について、表面からの任意のポイントの距離は、プレーン式係数を使用して計算することができる。

#### 【0266】

#### 【数49】

$$Dist = \frac{Ax + By + Cz + D}{\sqrt{A^2 + B^2 + C^2}}$$

係数が事前標準化されているとき、これを簡略化すると、

#### 【0267】

【数50】

$$Dist = Ax + By + Cz + D$$

となる。この計算は、増分で行われ得るので、任意のベクトルに沿って進むときに、距離は定数分だけ変化する。たとえば、1つのプレーンからの距離が位置[x, y, z]でDistである場合、X方向における1単位距離進めることは、距離を次のように変える。

【0268】

【数51】

$$\begin{aligned} Dist' &= A(x + 1) + By + Cz + D \\ &= Ax + By + Cz + D + A \\ &= Dist + A \end{aligned}$$

一般的に、任意のベクトル $r = [r_x, r_y, r_z]$ に沿って進む場合、プレーンからの距離は、

【0269】

【数52】

$$Dist' = Dist + r \odot [A, B, C]$$

ここで、

【0270】

【数53】

⊙

はドット・プロダクトを示す。三角形のバウンディング・ボックスを走査する間、三角形のプレーンからの距離は、1ボクセル当たり一回だけの加算で増分計算され得る。プレーンから距離を計算するために本発明に従って実施されるこの方

法は、図41に示す好ましい擬似コード・ルーチンによって示されている。

#### 【0271】

Y段階は、X段階よりも複雑である。それは、Y方向において1単位分進むばかりでなく、X方向において複数単位で後に進むからである。類似点として、リターンキーを一回押すと、紙の左マージンへ滑り戻り、行を進めるタイプライタの動作が考えられる。同様に、Z段階は、X、Y両方向において後に進むことと、Z方向において1単位分前方に進むことを結合する。この単純な前処理段階は、ボリューム全体にわたって効率的な進みを確実にする。数値的な近似問題が生じる場合には、各内側ループの開始時に距離値を記憶し、終了時にそれを復元し、それによって、内側ループ内の丸めエラーによる数値的なクリープを最小限に抑えることができる。

#### 【0272】

多価値ボクセル化の場合、7つのプレーン距離が必要である。したがって、プレーン距離を計算するのに1ボクセル当たり7回の加算が必要となる。1ボクセル当たりの他の計算としては、ループ・インデックスを増分させること、適切な領域を決定するための比較、および、必要に応じて、密度を決定する計算がある。

#### 【0273】

再び図40を参照して、領域R1において、1つのボクセルの密度値は、好ましくは、プレーンaに垂直な向きとなっているボックス・フィルタで計算される。プレーンからの距離DistAを与えた場合、密度値Vは、次の式を用いて計算される。

#### 【0274】

#### 【数54】

$$V = 1 - \frac{|DistA| - T/2}{W}$$

領域R2において、密度は、好ましくは、プレーンa、bからの距離を使用して

計算される。

【0275】

【数55】

$$V = 1 - \frac{\sqrt{\text{Dist}A^2 + \text{Dist}B^2} - T/2}{W}$$

同様に、領域R3は、プレーンa、cを使用し、領域R4、プレーンa、dを使用する。領域R5は、角隅ポイントC<sub>1</sub>からピタゴラス距離を使用する。

【0276】

【数56】

$$V = 1 - \frac{\sqrt{(C_1^x - x)^2 + (C_1^y - y)^2 + (C_1^z - z)^2} - T/2}{W}$$

同様に、領域R6、R7は、それぞれ、角隅ポイントC<sub>2</sub>、C<sub>3</sub>を使用する。

【0277】

隣接した三角形の共有エッジのところで、不連続またはクラックを回避すると好ましい。幸いにも、ボクセル化された表面のユニオンが正しく計算されたならば、方向付けられたボックス・フィルタは、任意の多面体についてのエッジの正確なフィルタリングを保証する。ユニオン演算子は、

【0278】

【数57】

$$V_{A \cup B} \equiv \max(V_A(x), V_B(x)).$$

での多価値密度値V(x)にわたって定義され得る。他のブール演算子も利用できる。しかしながら、max演算子は、共有エッジでの正しい向きのボックス・フィルタ値を保存するので、好ましい。

## 【0279】

本発明の方法においてmaxを使用することの含意は、現ボクセル値をメモリから読み出し、次いで、おそらくは、修正し、メモリに書き戻さなければならないということである。したがって、1ボクセル当たり最大2つのメモリ・サイクルが必要である。

## 【0280】

本発明のアルゴリズムの効率は、不必要な計算量を制限することによってさらに向上させることができる。これは、バウンディング・ボックスが、しばしば、三角形によって影響を受けるよりも三角形によって影響されないボクセルをより高いパーセンテージで含むからである。エッジ長が所定定数を上回るとき、バウンディング・ボックスは、三角形を再帰的に細分することによってより緊密にされ得る。

## 【0281】

混在するポリゴンおよびボリュームを視覚化するために、ポリゴンは、好ましくは、目標ボリュームにボクセル化され、単一のパスでレンダリングされる。ポリゴンがボリュームに関して移動する場合、ボクセル化は、データを壊さないようにオリジナルのボリュームのコピーに生じなければならない。多価値ボクセル化されたポリゴン・ボクセルは、ボリューム・データから区別するためにタグを付けられてもよい。こうして、ポリゴンを着色し、他のデータから別個に陰影を付けることができる。

## 【0282】

上記の好ましい三角形ボクセル化アルゴリズムは、本発明のCube-5ボリューム・レンダリング・システムの分散パイプラインにおいて能率的に実施される。このアルゴリズムは、ほんの少量のハードウェアを既存のパイプラインに加え、相互作用率で正確な多価値ボクセル化を実施する。請求項記載のCube-5ボリューム・レンダリング・アルゴリズムの1つの重要な利点は、ボリューム・データを決定論的オーダーで筋道正しくアクセスするということである。この特徴は、このアルゴリズムについてバウンディング・ボックスの規則正しいスキヤニングを可能にする。

## 【0283】

図42において、ここには、本発明によるボクセル化パイプライン全体の好ましい実施例が示してある。オンザフライ・ボクセル化が重要であるならば、本発明のシステムは、好ましくは、ボリューム・レンダリングおよびボクセル化のための別々のパイプラインを含んでいてもよい。ボクセル化が別のパスで生じた場合、これらのボリューム・レンダリングおよびボクセル化パイプラインを結合し、ボクセル化パイプラインが、ボリューム・レンダリング・パイプラインからの大部分のハードウェアを再使用してもよい。各三角形についてのセットアップは、好ましくは、2Dラスタイゼーションのためにセットアップがホスト・システム上で実施されると同様の方法で、ホスト・システムで行う。

## 【0284】

図42を参照して、パイプラインの第1ハードウェア・ステージ302において、7つのプレーンからの距離が計算されると好ましい。7つの単純な距離単位が、7つのプレーンの各々についての4つのレジスタに割り当てられる。好ましくは、1つのレジスタが、プレーンからの現距離を保持し、他の3台のレジスタが、はX、Y、Zの段階についての増分を保持する。図43は、本発明の好ましい実施例に従って形成された、7つのプレーンのうちの1つのための距離計算ユニット310を示している。この距離計算ユニット310は、パイプラインの距離計算ステージ302の一部として内蔵されてもよい（図42参照）。他の6つのユニットは、本質的に同じ設計であるが、異なる値を保持することができる。ボクセル化の各クロック・サイクル中、パイプラインは、好ましくは、X、Y、Zの方向のいずれかにおいて進み（すなわち、X段階312、Y段階314またはZ段階316を実施し）、それによって、運動の方向に従って現距離を更新する。ボリュームを通してルーピングするためのハードウェアは、既にボリューム・レンダリング・パイプラインに存在しており、したがって、三角形のバウンディング・ボックスを走査するためにここで再使用される。

## 【0285】

7つのプレーン距離を計算した後、結果の値は、好ましくは、パイプラインを流下する。図42に示すように、次のパイプライン・ステージ304が、好まし

くは、現ボクセルがどの領域に存在するかについて決定する。領域選択ステージ304の好ましい実施例においては、いくつかのケースの交互禁制により、真理値表の結果を決定するのに、7つだけのコンパレータが必要である。たとえば、プレーンbの負側（下側）から見た図40において、プレーンeからの距離に応じて、プレーンfまたはgからの距離をテストする必要はない。

#### 【0286】

図42を続けて参照して、領域を決定した後、次のパイプライン・ステージ306がフィルタ機能を計算する。現ボクセルが三角形のSボクセル単位以内にある場合、好ましくは、パイプラインのフィルタ計算ステージ306が起動されるだけである。さもなければ、現ボクセルは、本質的に三角形によって影響を受けず、異なった領域は、単純なリニア表示から複雑なピタゴラス距離評価までの範囲で種々の計算を必要とする。ハードウェアが、理想的には、すべてのケースを等しく取り扱わなければならないので、このようなハードウェアが、限定解像度ルックアップ表（LUT）によって平方根近似を実施できると好ましい。しかしながら、入力、出力の範囲は小さく、したがって、必要なLUTのサイズは小さくなる。さらに、本発明のCube-5ハードウェアは、ボクセル化のために再使用され得るボリューム・レンダリングに利用できるいくつかのLUTを有する。表示

#### 【0287】

#### 【数58】

$$V = 1 - (\sqrt{Dist} - T/2)/W$$

を計算するのに3つの別個のユニットを設ける代わりに、1つのLUTにすべての計算を任せるとより効率的である。この場合、入力は[0, 12]について定義された $Dist^2$ であり、出力は、範囲[0, 1]における密度値Vである。

#### 【0288】

7つの領域の交互禁制により、最も複雑なフィルタ計算だけのハードウェアを設けると充分である。図40を参照して、最も複雑な計算は、領域R5、R6、R7の角隅距離計算であり、好ましい実施例では、この計算は、前述の平方根L



U Tに加えて、5つの加算器と3つの乗算器を必要とする。領域R 2、R 3、R 4におけるライン距離計算はより単純であり、1つの加算器と2つの乗算器と平方根L U Tだけが必要である。領域R 1は、L U Tへの必要な入力である二乗距離を得るのに一回の乗算を必要とするだけである。

#### 【0289】

再び図42を参照して、パイプラインの最終ステージ308は、好ましくは、現ボクセル値および計算した密度評価を使用してmax演算を計算する。本発明の好ましい実施例において、max演算子は、単にマルチプレクサに取り付けたコンパレータであって、2つ値のうち大きい方の値を守りに書き戻すようになっている。バウンディング・ボックス内の大部分のボクセルが三角形に影響を与えるほど接近していないので、メモリ・バンド幅は、必要なボクセルを読み込むだけなので節減されることになる。さらなるバンド幅節減は、現ボクセル値を変えるこれらのボクセルをメモリに書き戻すだけで達成できる。メモリにワードをリクエストすることとメモリからワードを受け取ることとの間に若干の待ち時間があるので、ボクセルは、好ましくは、パイプライン内にできるだけ早く取り込み、結果を、メモリが受け取りまで待たせる。最終ステージ308は、書き戻しメモリであり、これは、依存性の心配なしにバッファされる。

#### 【0290】

横断を複雑にするスキューイングのコンテキスト外で本発明を説明してきた。しかしながら、本発明は、Y段階、Z段階距離更新値にスキューイングを構築することも意図している。スキューイングは、また、より多くの複雑さを本発明のCube-5ハードウェアに加える。具体的には、最も左のボクセルをY方向に1単位移動させ、それをバウンディング・ボックスの外に置くとき、パイプラインは、実際には、X方向においてp-1段階を行ってボクセルをバウンディング・ボックス内に保持する。同様に、最も左のボクセルがZ方向において1段階移動するとき、それは、また、負のX方向において1段階移動する。これは、前と同じ方法で取り扱われる。したがって、本発明の装置は、好ましくは、パイプラインが現に最も左のボクセルを処理しているとき、それを決定するのに14より多いレジスタと対応するロジックを加えることによってスキューイングを実施し

するようになっている。

#### 【0291】

プレフィルタリング（本発明のボクセル化方法と組み合わせて実施してもよい）を用いて、一連の異なった解像度のボリュームを最適に生成することができる。この技術は、異なったサイズの画像をレンダリングするのに有用である。ボリュームのサイズは、好ましくは、最終的な画像のサイズと一致するように選ぶ。こうして、エイリアシングが、すべての画像解像度で避けられ、画像スケールで見えないシーンの部分をレンダリングするという不必要な仕事の実施されない。

#### 【0292】

プレフィルタリングは、さらに、模型運動ぼけにも使うことができる。たとえば、オブジェクトがカメラを通り過ぎるとき、シャッタが開いている間、複雑なボリュームを走査することになり、これが運動ぼけを生じさせる。運動ぼけを正確にレンダリングするためには、普通のレンダリング技術は、複数の画像をレンダリングし、それを単一の画像にブレンドする。しかしながら、この方法は、非常に遅い。プレフィルタリングによれば、本発明は、ボクセル化中に一度走査動作を実施し、規則的なボリューム・レンダリングと同じ時間で運動ぼけをレンダリングする。この方法は、特に運動が一定である場合（たとえば、同じ方向への移動および／または回転）には良く作動する。たとえば、飛行中に安定した速度で回転しているヘリコプタのブレードを考える。たとえば、30Hzのアニメーション・フレーム率の場合に5Hzの率で回転しているブレードをボクセル化するためには、ブレードはフレーム毎に $5/30 \times (2\pi)$ の弧を描いて走査する。したがって、ブレードの外側エッジでは、密度値は非常に低くなり、ブレードは中心におけるよりも透明に見える。ここで、より小さいボリュームを走査すると、よりソリッドに見える。より低い密度値がより少ない不透明さを表し、より高い密度値がより多い不透明さを表すように、ボリューム・レンダリング転送機能をセットするとよい。

#### 【0293】

複数のボリュームメトリック・オブジェクトがオーバーラップするとき、ボリュームの投影画像は非常に複雑になる。たとえば、煙が雲から立ち昇るシーンを考え

られたい。明らかに、2つのボリュメトリック・オブジェクトは、最終フレームで結合された画像とは別個に、レンダリングすることはできない。したがって、本発明の1つの形態によって実施される好ましい方法では、複数のオブジェクトは、最終レンダリング・パスについて1つのオブジェクトに結合され、そこから画像を創り出す。

【0294】

2つまたはそれ以上のオブジェクトが同じスペースを占めるとき、各オブジェクトからのカラーは、投影サイト・レイに沿って各サンプル場所のところで一緒に変調されると好ましい。したがって、各オブジェクトを分類、陰影付け仕手から結合し、その後、カラー変調を行うことができると好ましい。あるいは、ボクセル・データが最初に結合された場合には、可能性のある組み合わせ毎に新しい転送機能が必要となろう。したがって、この後者の方法は好ましくない。

【0295】

本発明の1つの形態によれば、複数のオーバーラップしているボリュームを混合する好ましい方法は、各オブジェクトのスライスがインタレースされるように第1のオブジェクトのzディメンジョンにおいて第1オブジェクト以外のすべてを再サンプリングする。これは、すべてのオブジェクトを整合させる分類、シェーディング、変換を含む。オブジェクト変換は、並進、スケーリングを含み、これらは、好ましくは、最も近い隣接のコネクションおよび回転を用いて本発明の装置によって実施される。回転は、好ましくは、上述の本発明の回転方法を用いて実施される。

【0296】

互いに関して位置あるいは向きを変えないオブジェクトを含むシーンの場合、本発明は、ハイレベル・シーン・グラフ・コンパイルのような最適化を意図している。たとえば、静止オブジェクトは、一度結合してから、後のレンダリングのために保存されると好ましいが、非静止オブジェクトは、他のオブジェクトに関して移動する毎に再結合される。

【0297】

テクスチャ・マッピングが、高品質の画像効果（たとえば、表面ディテール、

均一照明、シャドウ)をシミュレーションする広範囲に使用されている技術である。一般的に言えば、テクスチャ・マッピングは、三次元(3D)表面上へ二次元(2D)画像をマッピングすることを含む。テクスチャ・マッピングが行われる間に、ジオメトリック・オブジェクトがスクリーン上へラスタライズされる。

( $x$ 、 $y$ )ピクセル座標は、好ましくは、( $u$ 、 $v$ )テクスチャ座標にマッピングされ、RGB $\alpha$ 値がカラー値として返されてスクリーン上のピクセル使用する。

#### 【0298】

基本的には、テクスチャ・マッピングに関係しているプロセスは2つある。すなわち、( $x$ 、 $y$ )座標から( $u$ 、 $v$ )座標へのマッピングと、どんなRGB $\alpha$ が所与の( $u$ 、 $v$ )座標と一致するかについての画像へのルックアップである。( $x$ 、 $y$ )座標から( $u$ 、 $v$ )座標へのマッピングは、好ましくは、当業者にとって明らかのように、単純なマトリックス乗算を含む。しかしながら、RGB $\alpha$ 値を返すように( $u$ 、 $v$ )座標の画像へのルックアップは複雑である。テクスチャ・ルックアップについての超大規模集積回路(VLSI)ハードウェア要件は、普通、かなりのコストで、現今のグラフィックボードの大きい部分を要求する。これは、( $u$ 、 $v$ )座標が、希に、個別の画像座標(テクセルと呼ばれる)に直接マッピングするという事実により、本来のものである。したがって、隣接したRGB $\alpha$ 値は、好ましくは、線形に補間され、正確な( $u$ 、 $v$ )座標でRGB $\alpha$ 値を生成する。

#### 【0299】

二次元(2D)補間は、一般的に、ピクセルが2つ以上のテクセルをカバーしない場合、充分である。しかしながら、マッピングが1テクセルより大きいピクセル範囲を生成する場合、アーティファクトが、2D補間方法を使用して画像内に導入される。高価なテクセル結合作業を避けるために、ミップ・マッピングと呼ばれる技術が、普通のグラフィックス・パイプラインによって利用され得る。ミップ・マッピングは、基本的には、画像の複数のディテール・レベル(LOD)を記憶することからなる。次いで、( $x$ 、 $y$ )ピクセルが、( $u$ 、 $v$ )テクセルにマッピングされるとき、適切なMip-Mapレベル・テクセルは、ピクセルがテクセルよりも小さくなるように選ばれる。より正確な方法は、ディテール・テクセルの高いレベル、低いレベルの両方から4つの隣り合うテクセルをルック

アップし、全部で8つのテクセルにトリリニア補間を実施してピクセルについて適切なRGB $\alpha$ 値を計算することである。

### 【0300】

普通のグラフィックス・パイプラインからのテクスチャ・マッピング・ハードウェアは、ボリューム・レンダリングを加速するために用いられてきた。これは、Reality Engine Graphics, by K. Akeley, Computer Graphics (SIGGRAPH93), 27:109116, Aug. 1993およびAccelerated Volume Rendering and Tomographic Reconstruction Using Texture Mapping Hardware, by B. Cabral, N. Cam and J. Foran, Symposium on Volume Visualization, pp. 91-98, Oct. 1994のようなテキストの主題であった。しかしながら、この普通の方法は、コスト・パフォーマンスを達成しなければ、本発明の種々の機能（たとえば、シェーディング）をサポートもしない。さらに、公知の従来技術方法を用いる場合、テクスチャ・マッピングは、データ複写なしにスケーリングすることができず、しばしば、三次元（3D）補間よりもむしろ二次元（2D）補間を使用し、データセットのダウンロードが遅く、および／またはリアルタイム4次元（4D）入力をサポートしていない。

### 【0301】

上記した本発明の好ましい形態によれば、Cube-5装置は、ジオメトリ入出力バス46、48（図4参照）を経て普通のジオメトリ・エンジンと結合される。好ましくは、本発明のレンダリング・パイプライン（単数または複数）は、テクスチャ・ルックアップ機能を実施するのに利用されると共に、ジオメトリ・エンジンが、(u, v) テクスチャ座標に (x, y) ピクセル座標をマッピングするのに使用される。簡単に言えば、ひとたびCube-5装置と結合したならば、ジオメトリ・エンジンの責任は、本質的に、三角形をラスタライズすることであり、一方、本発明の装置は、好ましくは、テクスチャ・マッピングのための高性能補間エンジンを提供する。本発明の装置上でテクスチャ・ルックアップを実施するためには、テクセル・データは、好ましくは、Cube-5ユニット（単数または複数）内に含まれる3Dメモリにロードされる。図6A、6Bは、2×2近傍についての32ビット・テクセル・データが、どのようにして、16ビッ

ト・ボクセルの $2^3$ サブキューブに配置されるかの一例を示している。

### 【0302】

本発明の重要な別の利点は、画像ベースのレンダリングを強化する能力である。一般に、画像ベースのレンダリング方法は、複雑なシーンを、このシーンの有限の画像セットに基づいて任意の視点からレンダリングする。当業者には公知の類似した画像ベース・レンダリング方法が2つあり、すなわち、明視野レンダリングおよびルミグラフィであり、これらの方法は、ソース画像のデプス情報を必要とすることなく4次元(4D)補間を用いる。

### 【0303】

図44は、明視野レンダリングにおいて、シーンが $uv322$ および $st320$ プレーンによってモデル化されることを示している。あらゆる $uv$ グリッド・ポイントが、好ましくは、視点を定め、関連した $st$ 画像を有する。投影プレーン $324$ のあらゆるピクセルについて、サイト・レイ $326$ が、好ましくは、 $uv$ プレーン $322$ にキャストされる。 $uv$ プレーンとサイト・レイとの交差を取り囲んでいる $uv$ グリッド・ポイントに対応する4 $st$ 画像が、そのレイに寄与する。この寄与は、好ましくは、各 $st$ 画像にその $uv$ グリッド・ポイントを通してサイト・レイをキャストすることによって計算される。これらのレイは、 $st$ 画像ピクセル間にヒットし、したがって、各 $st$ 画像についてバイリニア補間を実施しなければならない。4つのレイ間の1回の最終バイリニア補間で、最終投影プレーン・ピクセル・カラーを生成する。したがって、投影プレーン $324$ のあらゆるピクセルを得るには、 $st$ プレーン $320$ における4回のバイリニア補間と、 $uv$ プレーン $322$ における1回のバイリニア補間とを必要とする。その結果、全部で5回のバイリニア補間が必要である。これら5回のバイリニア補間は、実質的に、1つの4D補間または15のリニア補間に等しい。

### 【0304】

各投影プレーン・レイについてルックアップを実施することは、通常、 $st$ 画像へのランダム・アクセスを生じさせる。したがって、本発明の好ましい方法によれば、 $st$ 画像は、オブジェクト・オーダーでアクセスされる。このオブジェクト・オーダーは、Cube-5装置が一度だけ各 $st$ 画像ピクセルの読み出し

を許すので、本発明の装置と共に用いるのにより適っている。図44を続けて参照して、 $uv$ プレーンにおける各四辺形328（たとえば、 $abcd$ ）について、4つの $st$ プレーンへの四辺形の投影像（たとえば、 $abcd$ に対応するプレーン）は、好ましくは、4つのうちのどのタイル領域330が最終画像に寄与しているかを決定する。次に、すべての $st$ タイル領域330が、好ましくは、4つの画像に組み立てられ、投影プレーン324上へ透視投影される。4つの投影画像の中の最終画像は、続いて、バイリニア補間によって形成される。補間重みは、好ましくは、オリジナル・レイおよび $uv$ プレーン322の交差によって決定される。

### 【0305】

添付図面を参照しながら本発明の実施例を説明してきたが、本発明がこれらの実施例そのものに限定されるものではなく、本発明の範囲または精神から逸脱することなく種々の他の変形例、変更例が当業者によって実施可能であることは了解されたい。

#### 【図面の簡単な説明】

【図1】 普通のボリューム視覚化システムのブロック図である。

【図2】 本発明の一実施例に従って形成したユニバーサル三次元レンダリング・システムを示す概念ブロック図である。

【図3】 本発明の好ましい実施を示す図2のCube-5ユニットの簡略ブロック図である。

【図4】 本発明の一実施例に従って形成したユニバーサル三次元レンダリング・アーキテクチャの概要を表す機能的ブロック図である。

【図5】 本発明の一実施例に従って形成したユニット・レンダリング・パイプラインを示す機能的ブロック図である。

【図6A】 本発明の好ましい方法に従って、32ビットのテクセル・データをどのようにして16ビット・ボクセルのミニブロックに $2 \times 2$ 近辺について記憶するかを示すグラフ図である。

【図6B】 図6Aの例についてのボクセル記憶、テクセル記憶の比較表である。

【図7】 本発明の好ましい順方向画像ワーピング方法に従ってソース、目標画像に特殊な平行保存スキャンラインを示している。

【図8】 本発明の好ましい順方向画像ワーピング方法に従って平行保存スキャンライン方向を決定する方法を示すグラフ図である。

【図9】 本発明の好ましい順方向画像ワーピング方法に従ってスキャンライン処理を実施するためのソース画像におけるピクセル読み取りテンプレートを示す例の二次元グラフ図である。

【図10】 本発明の好ましい順方向画像ワーピング方法に従って、スキャンライン処理を実施するためのバイリニア・サンプル補間を示す、図9の例の二次元グラフ図である。

【図11】 本発明の好ましい方法に従って、目標ピクセル訂正を実施するためのピクセル値を得るサンプルについてのリニア補間の二次元グラフ図である。

【図12】 本発明の好ましい順方向画像ワーピング方法に従ってアンチエイリアシングを実施するための異方性フィルタ・フットプリントの計算を説明するグラフ図である。

【図13】 本発明の好ましい順方向画像ワーピング方法に従って、目標サンプル上へソース・ピクセルをスプラッチングことを説明するグラフ図である。

【図14】 本発明の1つ方法に従って、二次元スライス・シャープ分解によって三次元回転を実施するためのyスライス・シャープの例を示す。

【図15】 本発明の別の方法に従って、二次元ビーム・シャープ分解によって三次元回転を実施するためのxビーム・シャープの例を示す。

【図16】 本発明のさらに別の方法に従って、二次元スライス・ビーム・シャープ分解によって三次元回転を実施するためのxスライス-yビーム・シャープの例を示す。

【図17】 本発明のまた別の方法に従って、三次元ビーム・シャープ分解によって三次元回転を実施するための三次元xビーム・シャープの例を示す。

【図18A】 透視投影を実施する普通のアンダーサンプリング方法を示



す。

【図18B】 透視投影を実施する普通のオーバーサンプリング方法を示す。

【図19】 本発明の好ましい形態に従って、透視ボリュメトリック投影を実施する適応透視レイ・キャスト方法を示す図であり、ビュー切頭体を、視点から指数的に増加する距離に基づく領域に分割することを示す図である。

【図20A】 本発明の好ましい適応透視レイ・キャスト方法に従って、指数領域境界でのレイの分割／結合を示すグラフ図である。

【図20B】 図20Aの適応透視レイ・キャスト方法のレイ・セグメントA、B、Cについての効果的フィルタ重みを示すグラフ図である。

【図21】 本発明の好ましい形態に従って、サイズ±2サンプルの二次元フィルについての重みの例を示す。

【図22】 本発明の適応透視レイ・キャスト方法の例を示すグラフ図であり、 $7^3$  ボリュームが視点からの3つのボクセル単位であることを示す図である。

【図23】 本発明の1つ形態に従って、指数領域透視後前投影を実施する好ましい方法の擬似コード表現である。

【図24】 2つの領域を横切ったの、本発明の指数領域透視レイ・キャスト方法の例を示す。

【図25】 本発明の一実施例に従って、Sobel 勾配フィルタのx成分の $3^3$ の対称近似を実施するための好ましい重みの例を示す。

【図26】 本発明の一形態に従って単一画像においてジオメトリック・オブジェクトおよびボリュームを混合する方法を示すグラフ図である。

【図27】 本発明の1つの形態に従って三角形を薄いスラブ境界にクリップする方法のグラフ図である。

【図28】 本発明の好ましい形態に従って半透明のポリゴンをバケット・ソートする方法のグラフ図である。

【図29】 本発明の1つの形態に従って、ポリゴン・フットプリントをプレワープすることによってシャード視線ジオメトリを創作する方法のグラフ図

である。

【図30】 本発明の1つの形態に従って形成したCube-5パイプラインのグラフ図であり、そこに内蔵されるSDRAM合成バッファを示す図である。

【図31】 テクスチャ・メモリ、フレーム・バッファおよびジオメトリ・パイプライン間のインタフェースを概念的に示す、普通のグラフィックス・アクセラレータのグラフ図である。

【図32】 ジオメトリ・パイプラインを本発明のCube-5ボリューム・レンダリング・パイプラインと接続する二重用途DRAMフレーム・バッファを使用している本発明の一実施例を示すグラフ図である。

【図33】 本発明の1つの形態に従って、コクセルFIFO待ち行列を含む各Cube-5パイプラインについてのメモリ・インタフェースを示すブロック図である。

【図34】 本発明の好ましい実施例に従って形成した、8ドラム・チップ上のRGBAコクセル・レイアウトのグラフ図である。

【図35】 本発明の一形態に従って形成した、実行長コード化(RLE)フレーム・バッファ・ハードウェアの埋め込みDRAMチップの部分ブロック図である。

【図36】 本発明の1つの形態に従って、図35のRLEハードウェアで生じる処理を示す擬似コード表現である。

【図37】 Cube-5パイプラインに含まれるSDRAM合成バッファにジオメトリ・パイプラインを接続するRLEフレーム・バッファを示す本発明の好ましい実施例のグラフ図である。

【図38】 本発明の1つの形態に従って、表面に対して垂直方向外方の、ソリッド・プリミティブの中心からのラインに沿って採用した指向性ボックス・フィルタの密度分布を示す。

【図39】 本発明の別の形態に従って、三角形表面プリミティブに対して垂直なラインに沿って採用した指向性ボックス・フィルタの密度分布を示す。

【図40】 本発明の好ましい実施例に従って。三角形プリミティブのた

めの7つのボクセル化領域の二次元図である。

【図41】 本発明の1つの形態に従って、1つのプレーンからの距離を計算する方法の擬似コード表現である。

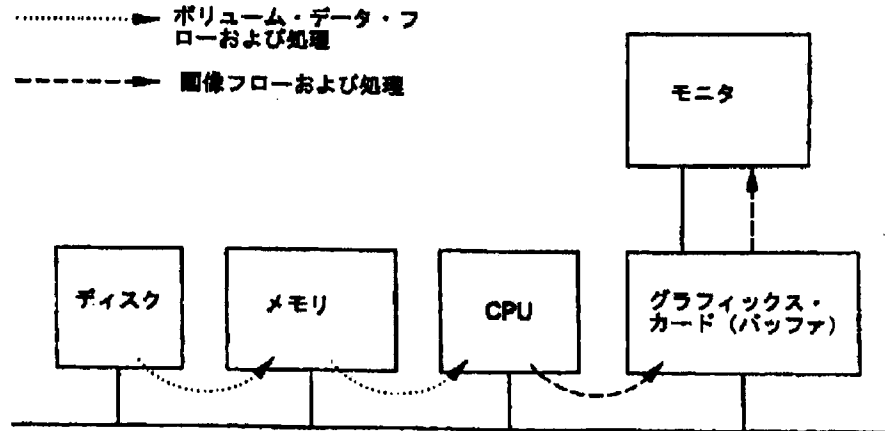
【図42】 本発明の一実施例に従って形成した、ハードウェア・ボクセル化パイプラインの概要を示すブロック図である。

【図43】 本発明の一形態に従って、所望のプレーンからの現ボクセルの距離を増分計算する距離ユニットを表すブロック図である。

【図44】 本発明の1つの形態に従って画像ベースのレンダリングを実施する好ましい方法を示す頂面グラフ図である。

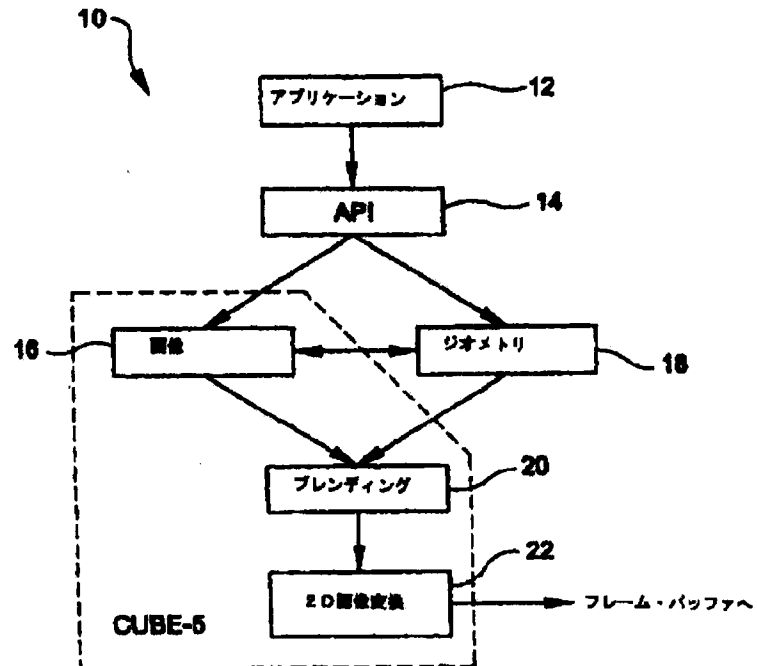
【図1】

**FIG-1** 従来技術



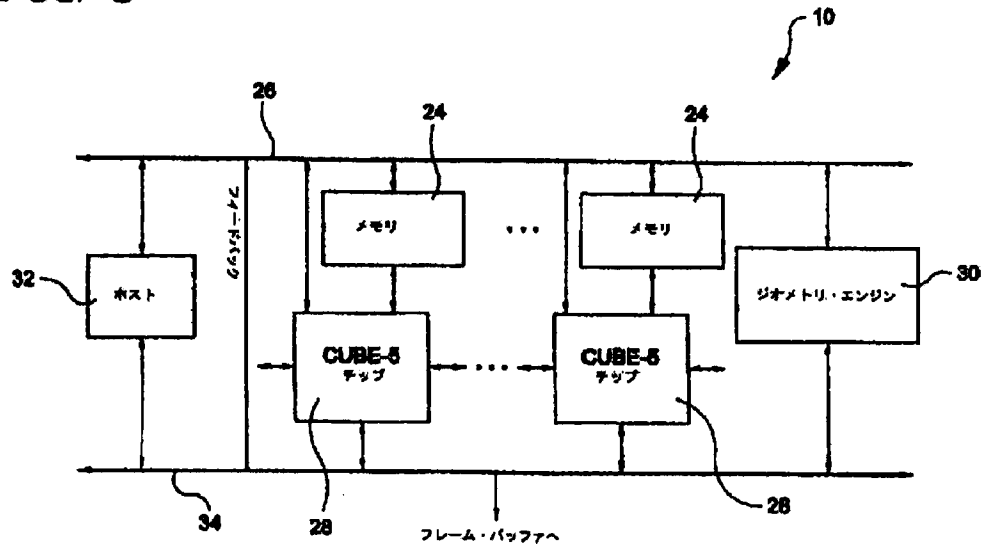
【図2】

FIG-2



【図3】

FIG-3



【図4】

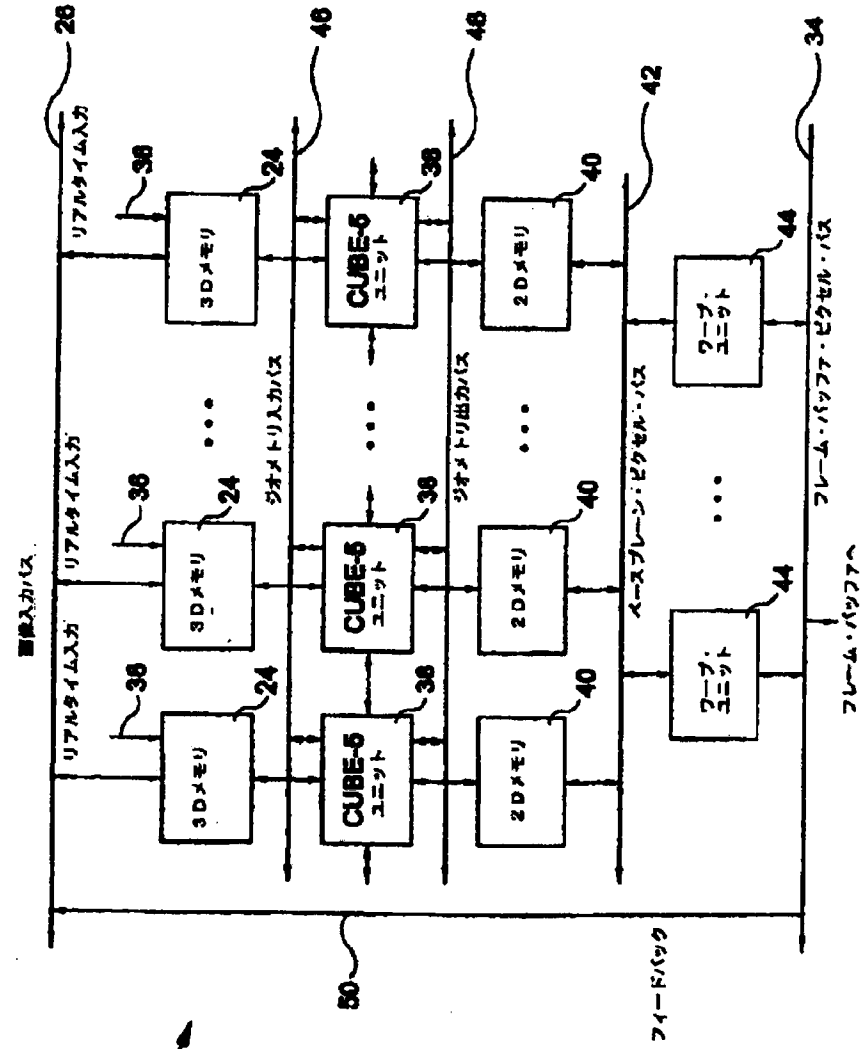
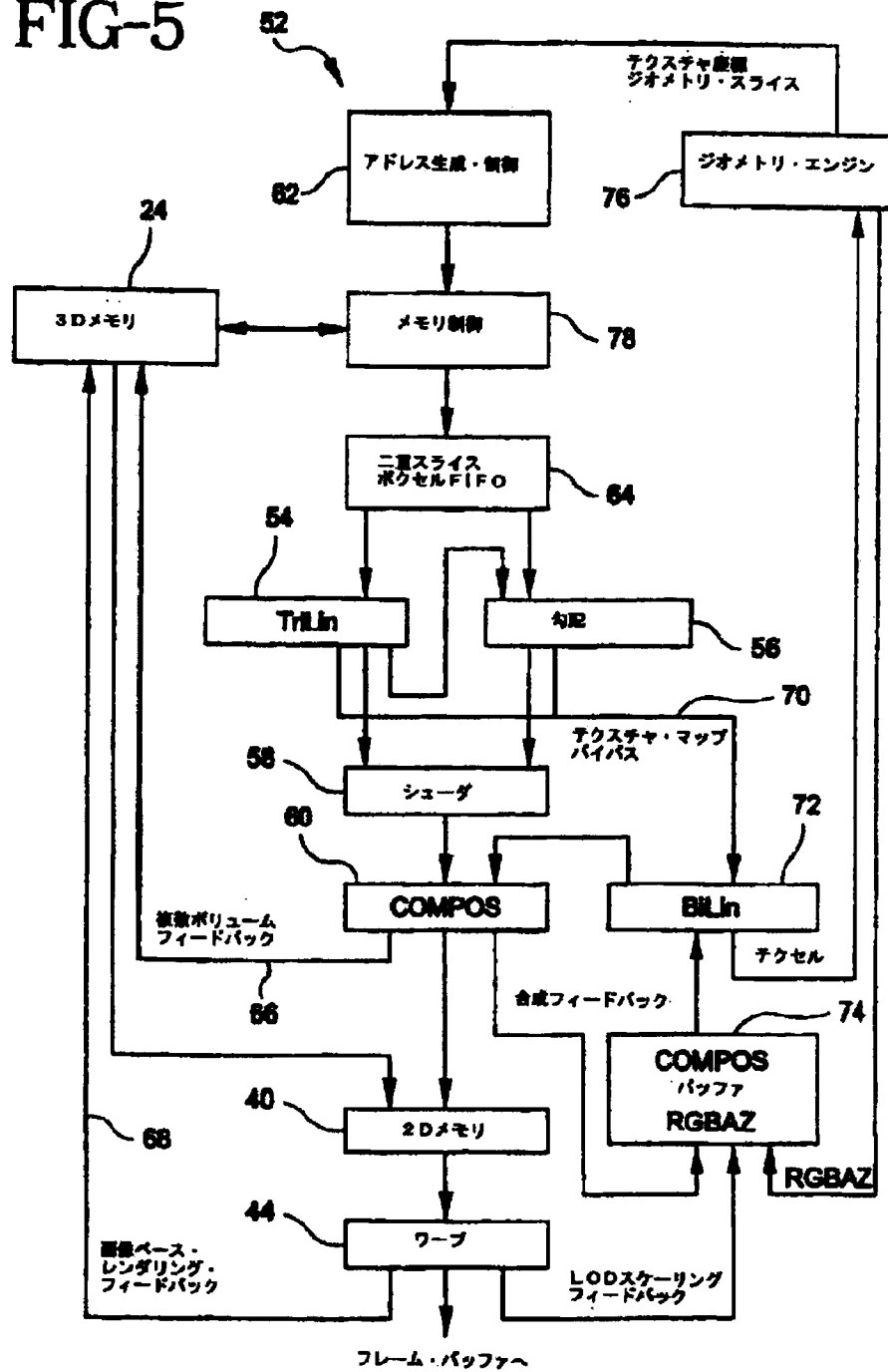
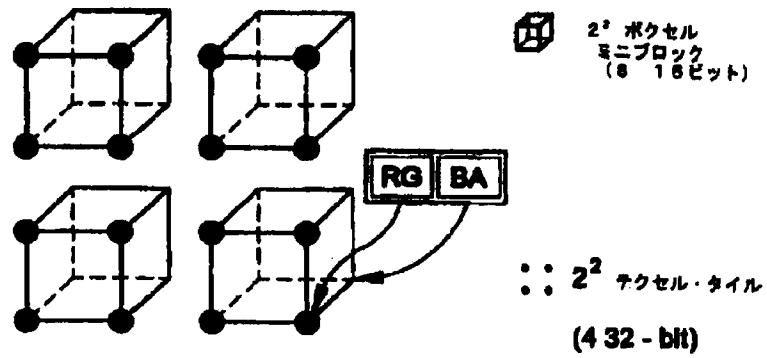


FIG-5



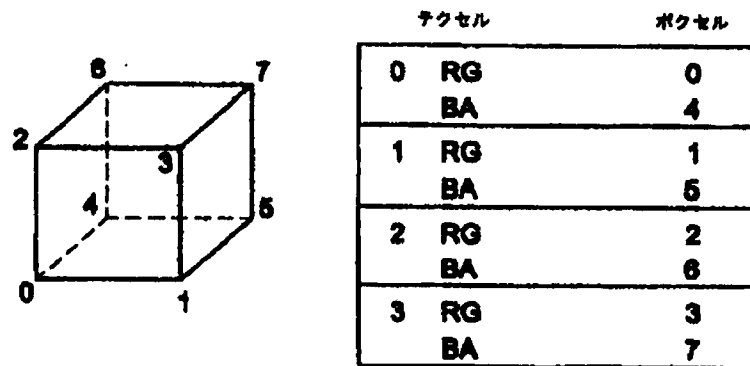
【図6A】

FIG-6A



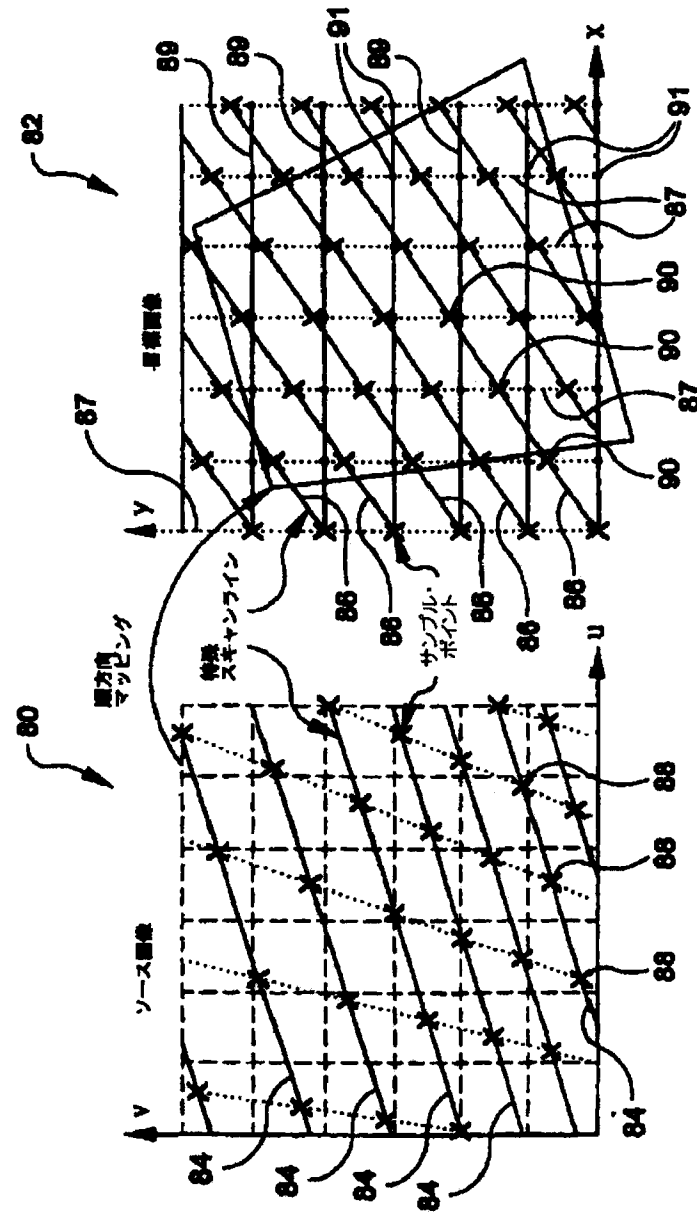
【図6B】

FIG-6B



【図7】

FIG-7





**FIG-8**

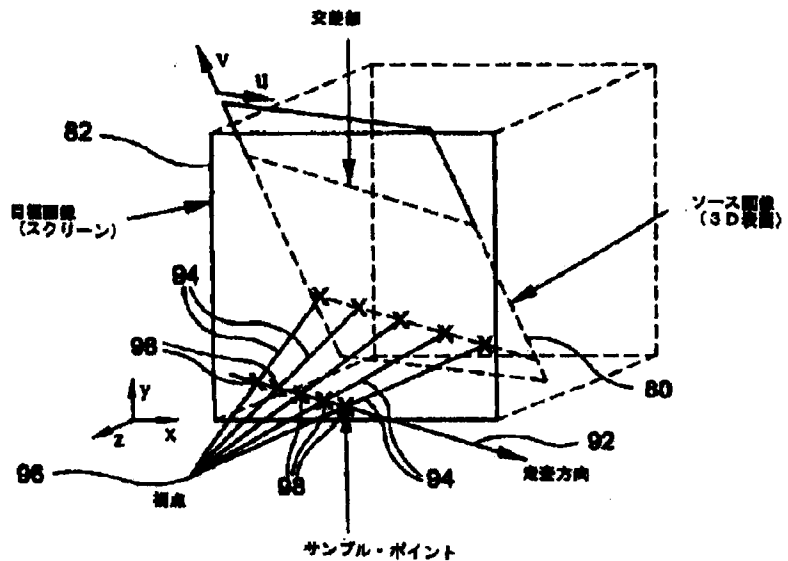
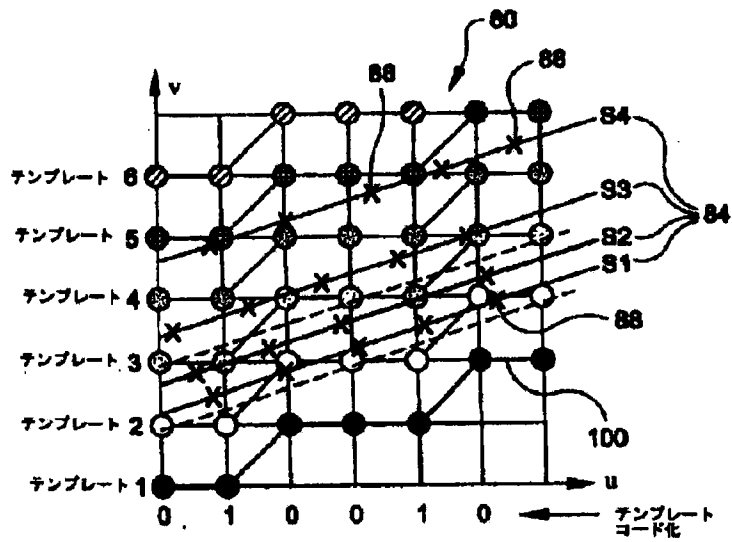
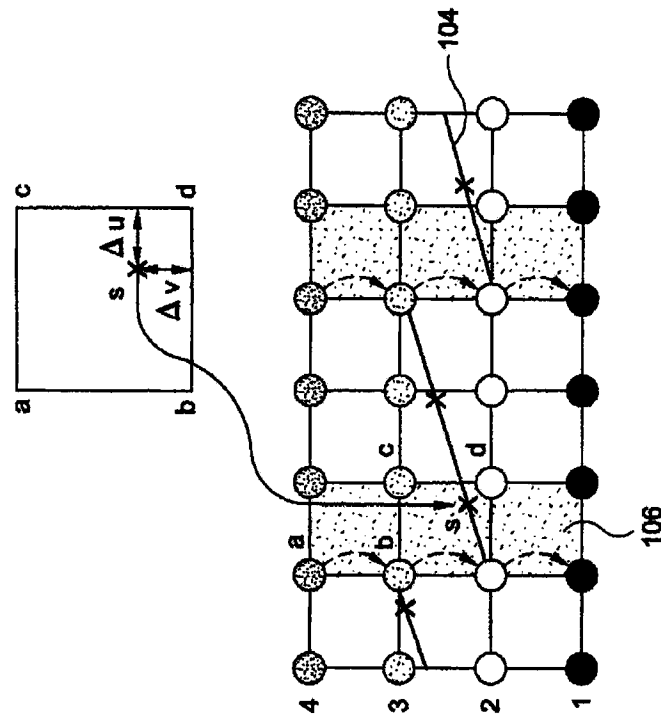


FIG-9



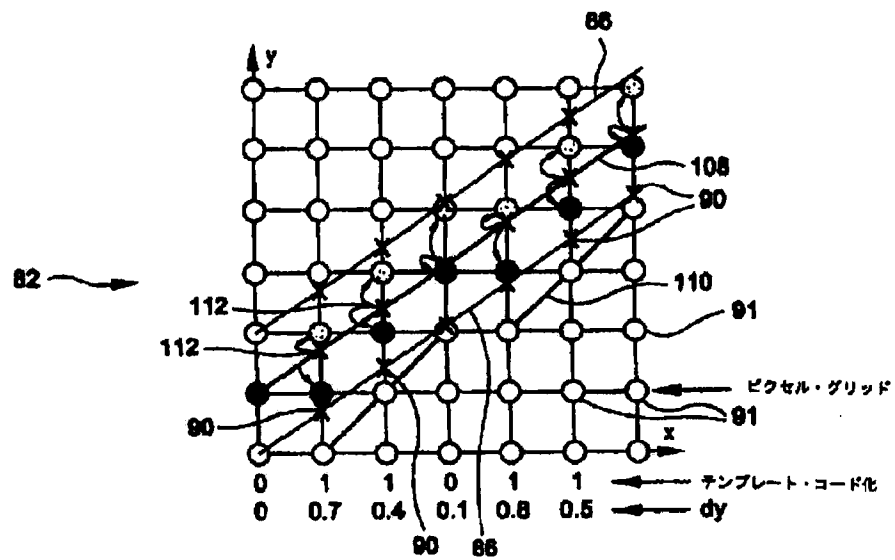
【図10】

FIG-10



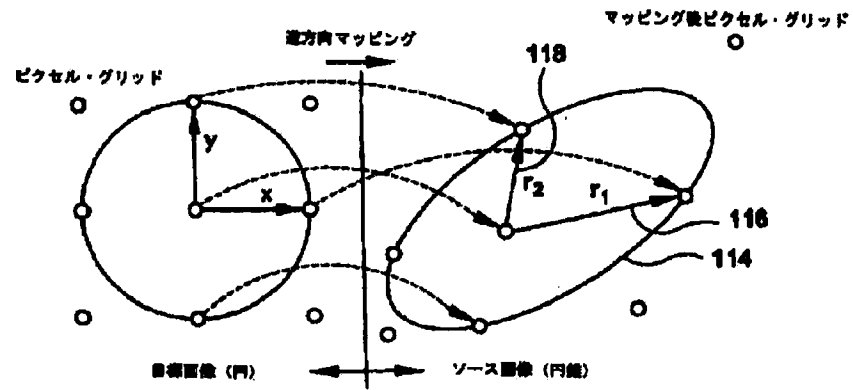
【図11】

FIG-11



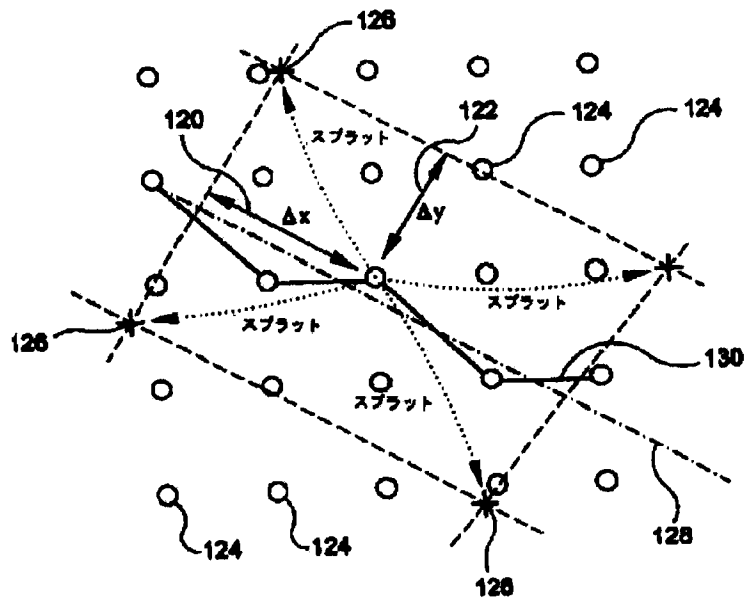
【図12】

FIG-12



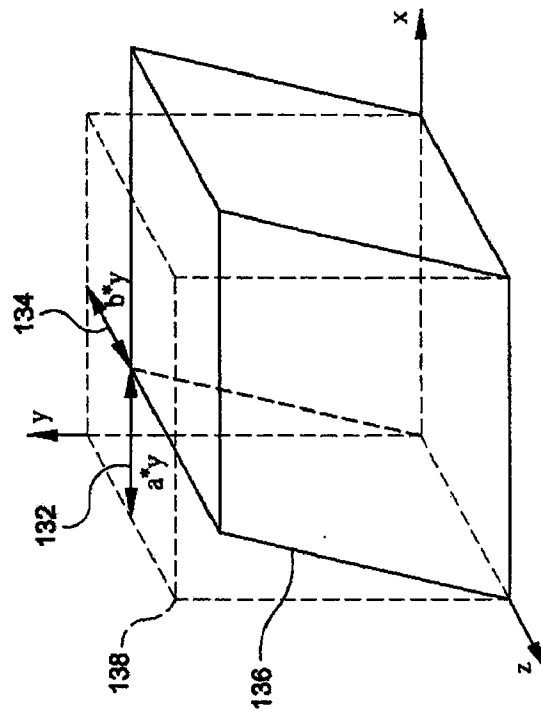
【図13】

FIG-13



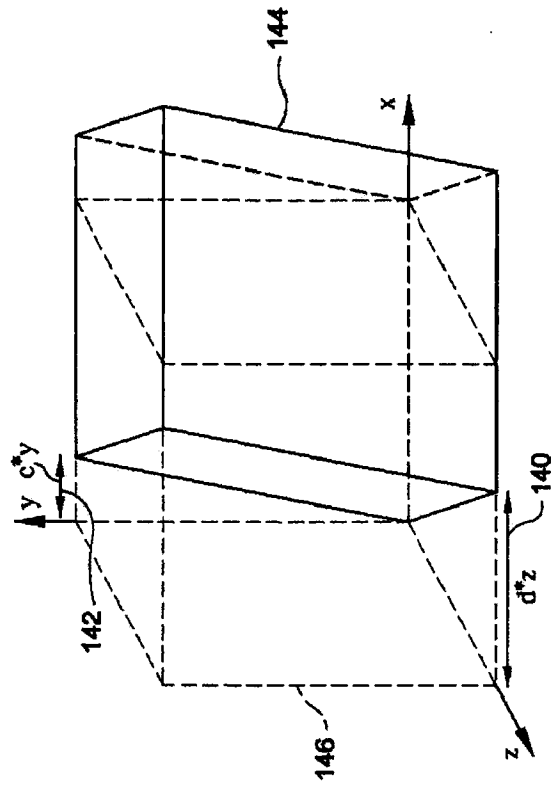
【図14】

FIG-14



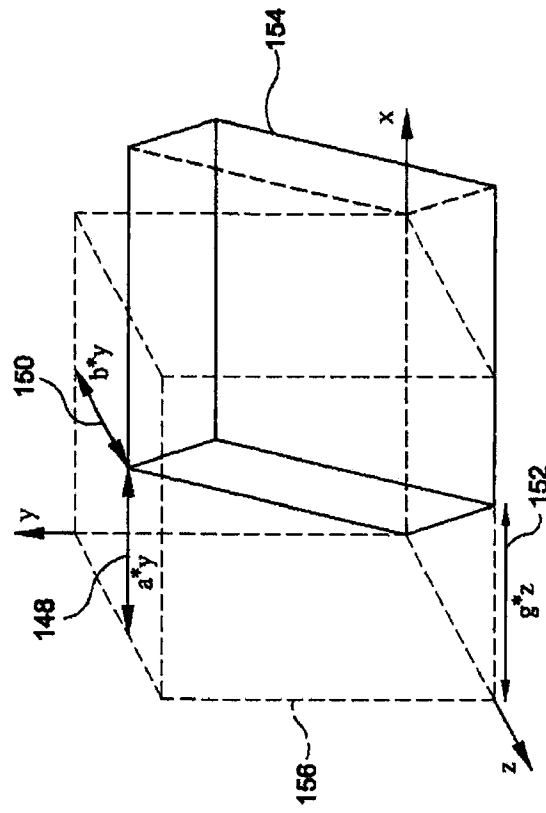
【図15】

FIG-15



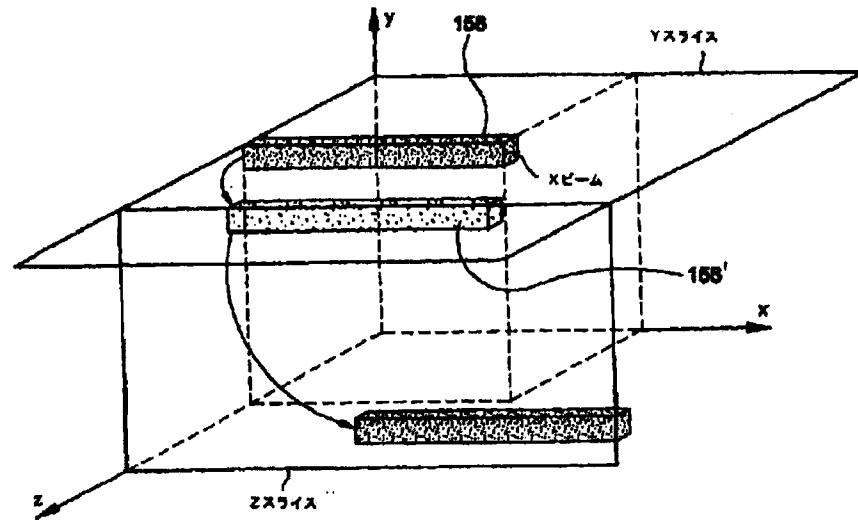
【図16】

FIG-16



【図17】

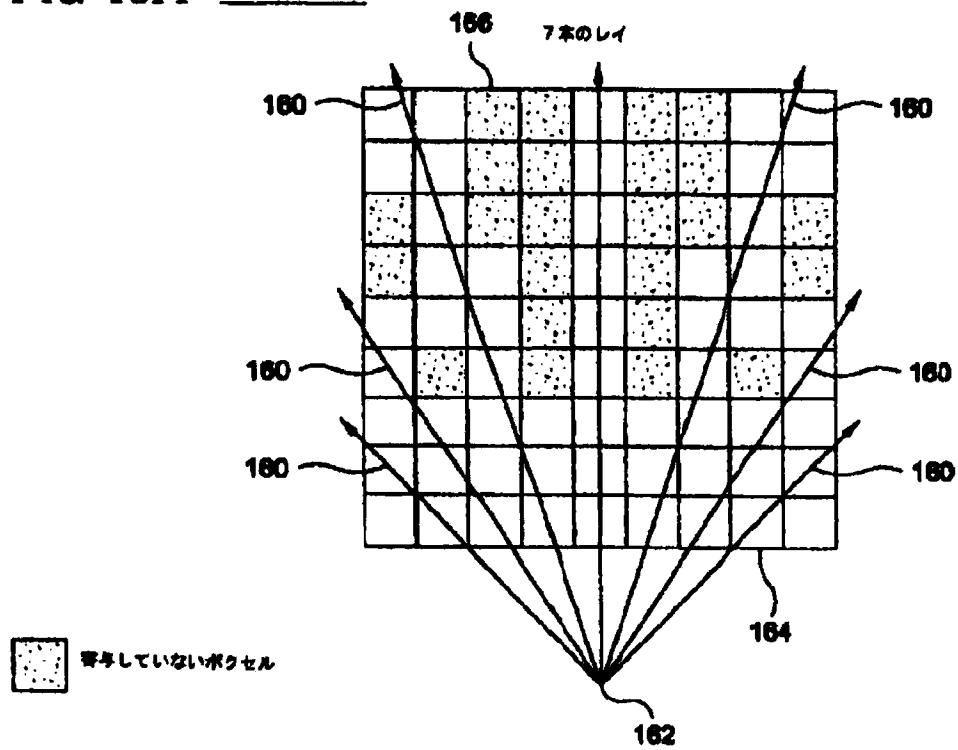
FIG-17



【図18A】

FIG-18A

従来技術

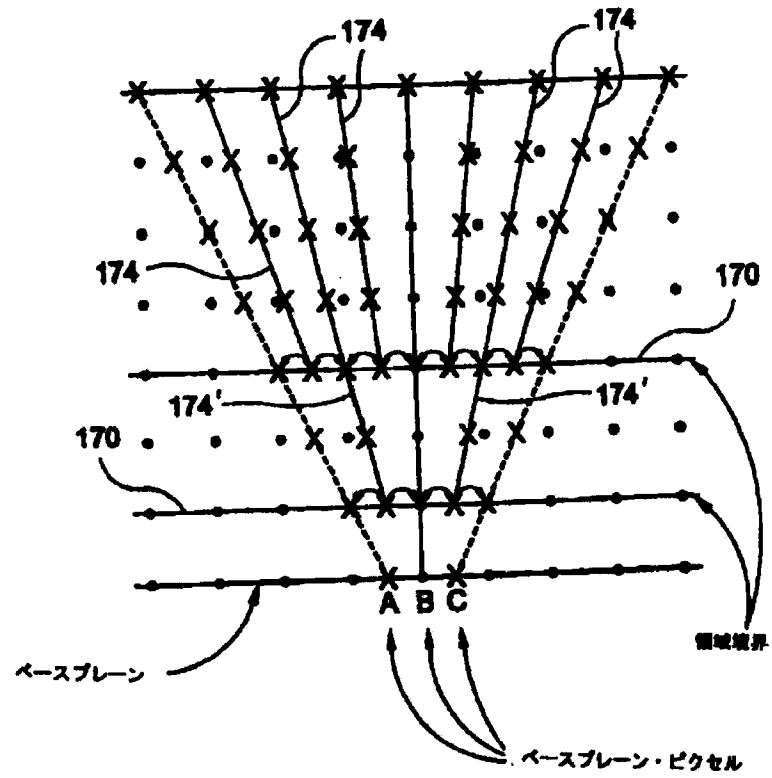






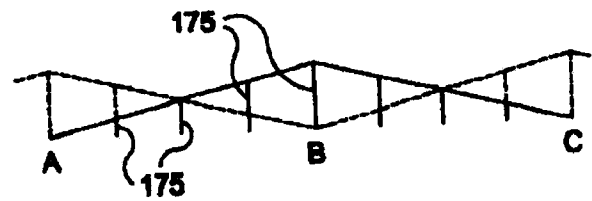
【図20A】

FIG-20A



【図20B】

FIG-20B



【図21】

FIG-21

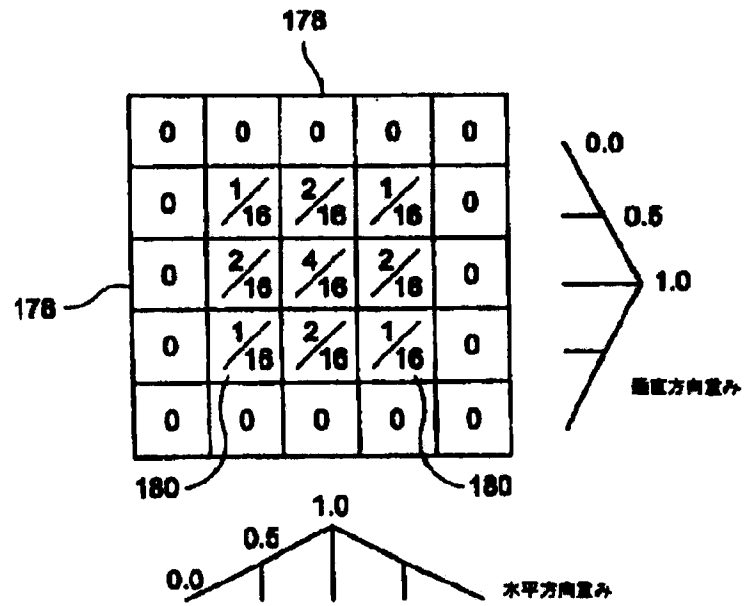
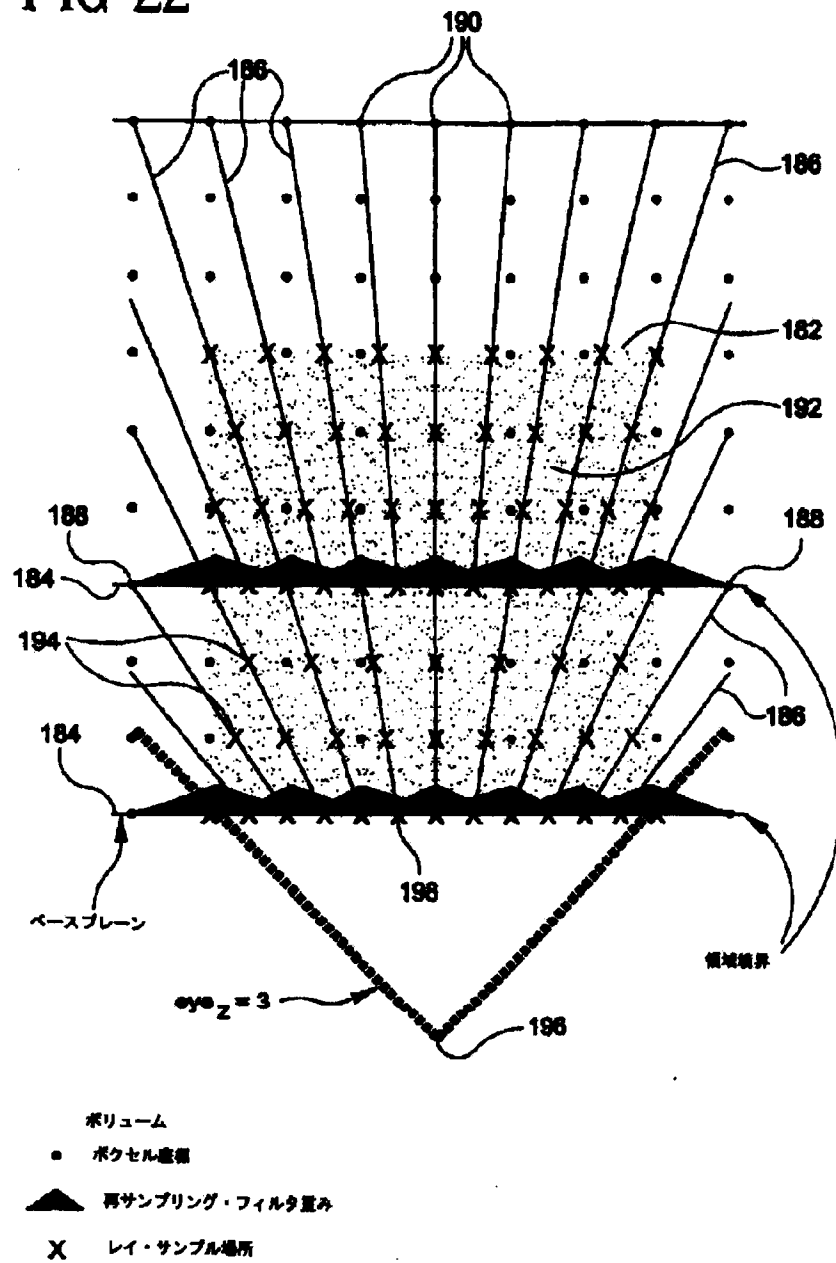


FIG-22



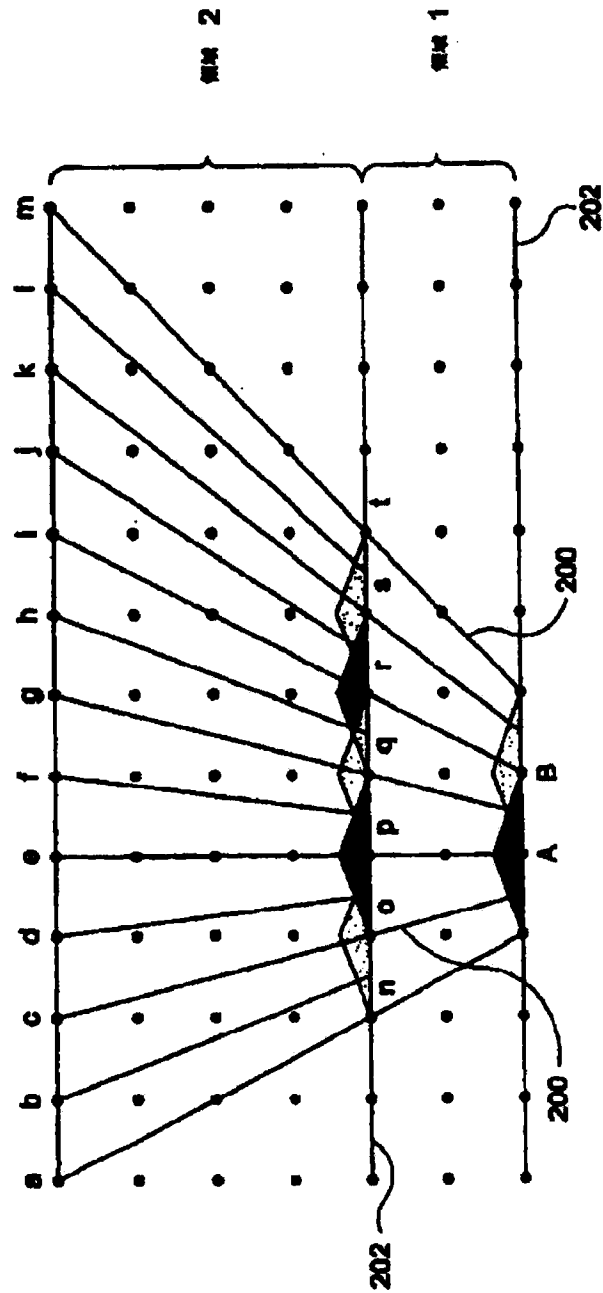
【図23】

FIG-23

```
Compute Z-position of Eye in Voxel Units
Compute Exponential Region Boundaries
for REGION = MAXREGION to 0
  for SLICE = MAXSLICE[REGION] to MINSLICE[REGION]
    Interpolate Samples for this slice (Bilin)
    Shade and Classify Samples
    Composite Samples onto Rays in Buffer
  end for
  if not frontmost REGION
    Downsample Rays in Compositing Buffer with
      Bartlett Filter
    end if
  end for
Warp Baseplane to Final Image plane
```

【図24】

FIG-24



レイザリングのための  
パトリック・フィルタ

【図25】

FIG-25

x重み: -1 0 1

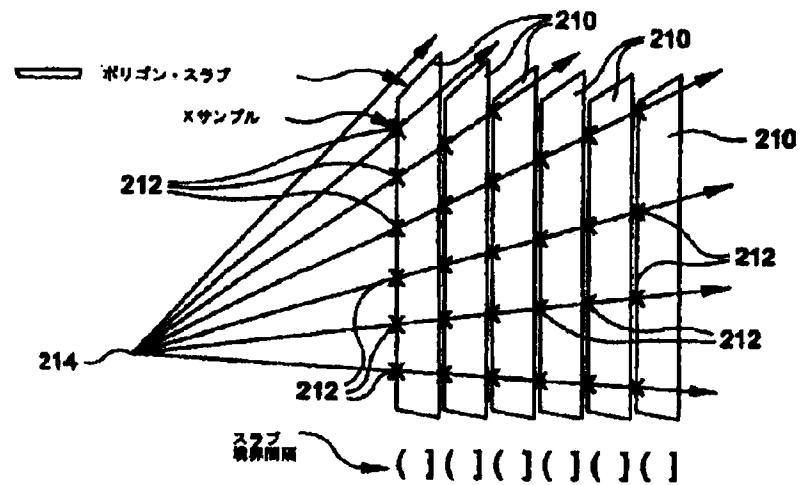
1  
y重み: W  
1  
1  
x重み: W  
1

生成された有効3×3×3フィルタ:

		-1 0 1
	-w 0 w	-w 0 w
-1 0 1	-w <sup>2</sup> 0 w <sup>2</sup>	-1 0 1
-w 0 w	-w 0 w	
-1 0 1		

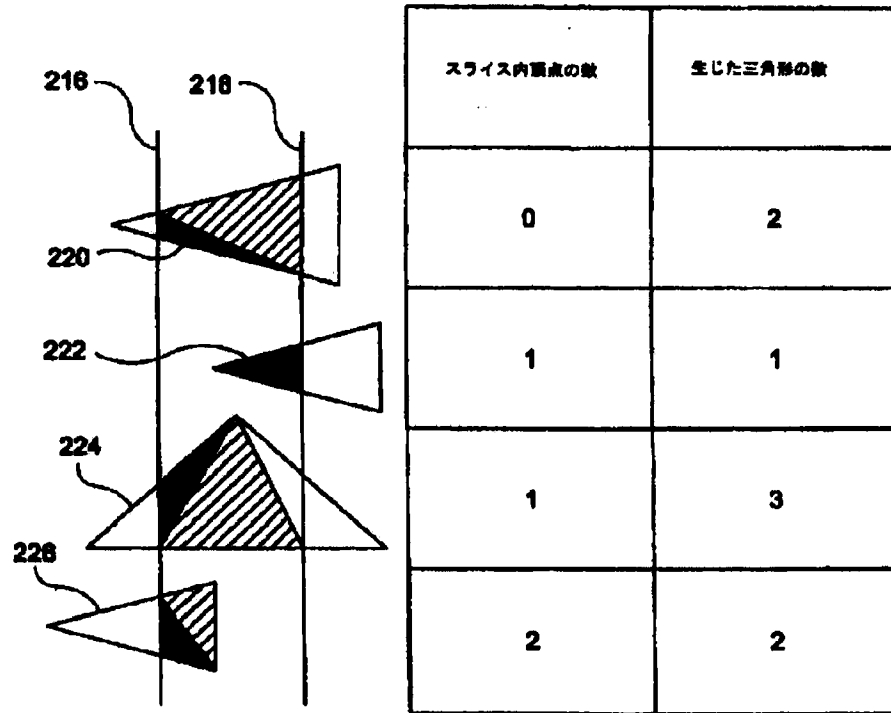
【図26】

FIG-26



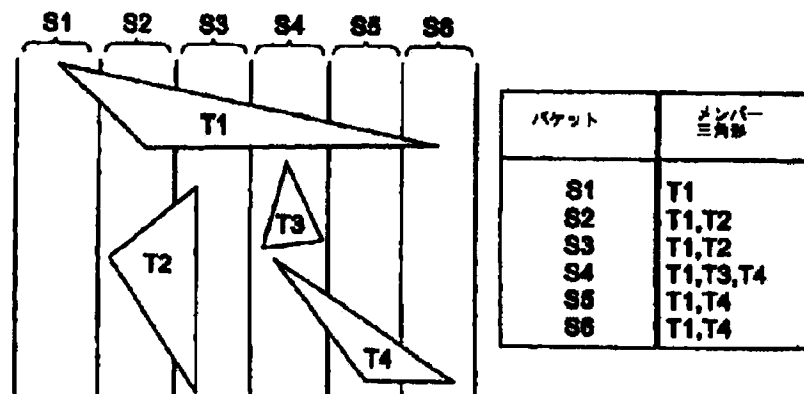
【図27】

FIG-27



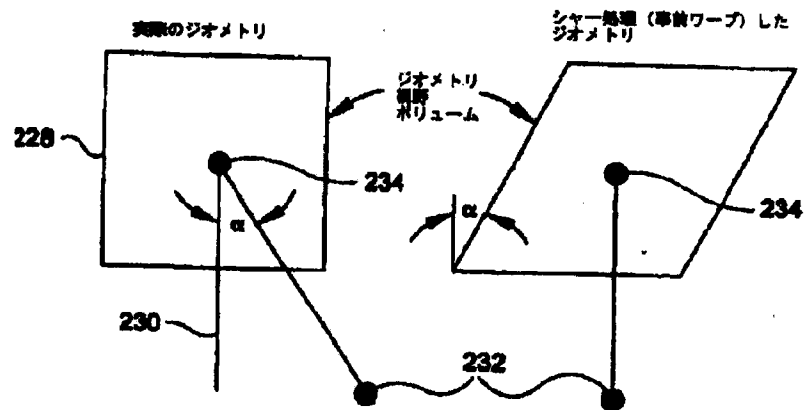
【図28】

FIG-28



【図29】

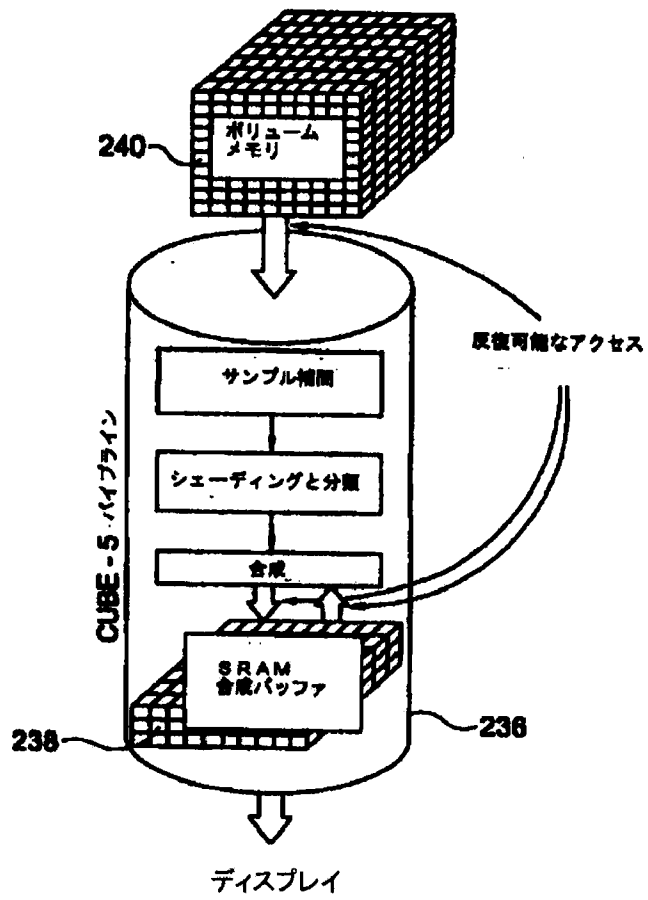
FIG-29





【図30】

FIG-30



**FIG-31** 従来技術

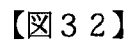


FIG-32

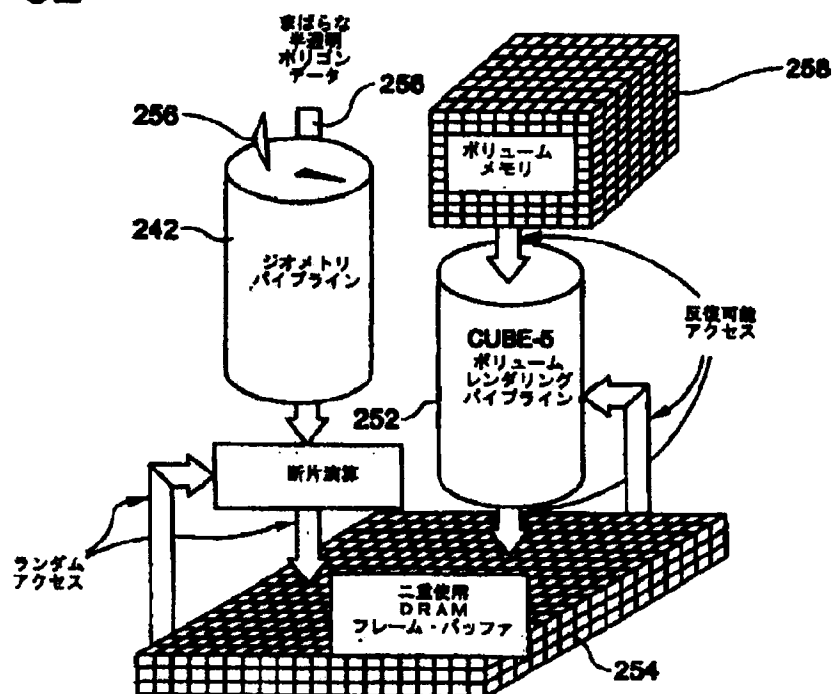


FIG-33

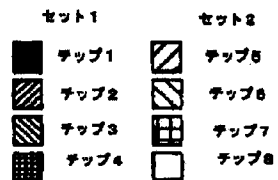
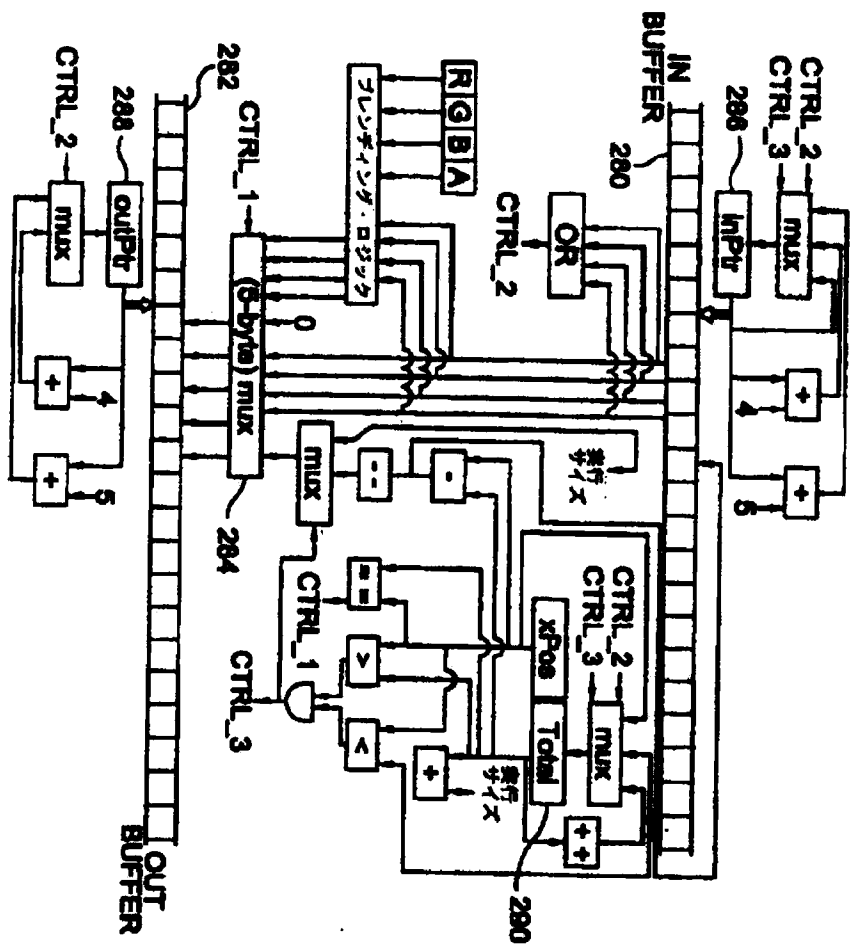


FIG-35



【図 35】

FIG-36

```

RLE_AddFragment(xPos, yPos, RGBA) {
    tmp = nextFreeScanline();
    RLE_AddPixelToScanline(data[yPos], xPos, RGBA, tmp);
    freeScanline(data[yPos]);
    data[yPos] = tmp;
}

RLE_AddPixelToScanline(in, xPos, RGBA, out) {
    total = 0;
    inPtr = 0;
    outPtr = 0;
    while(total < lineWidth) {
        if (total == xPos) {
            out[outPtr+3] = BLEND(RGBA, in[inPtr+3]);
            outPtr += 4;
            total++;
            if(in[inPtr+3] == 0)
                in[inPtr+4]--;
            else
                inPtr += 4;
        }
        out[outPtr+3] = in[inPtr+3];
        if (in[inPtr+3] == 0) {
            if (total < xPos && total+in[inPtr+4] > xPos) {
                out[outPtr+4] = xPos - total - 1;
                outPtr += 5;
                in[inPtr+4] -= xPos-total;
                total = xPos;
            } else {
                out[outPtr+4] = in[inPtr+4];
                total += in[inPtr+4];
                outPtr += 5;
                inPtr += 5;
            }
        } else {
            total++;
            outPtr += 4;
            inPtr += 4;
        }
    }
    out[outPtr+3] = in[inPtr+3];
    if (in[inPtr+3] == 0) {
        if (total < xPos && total+in[inPtr+4] > xPos) {
            out[outPtr+4] = xPos - total - 1;
            outPtr += 5;
            in[inPtr+4] -= xPos-total;
            total = xPos;
        } else {
            out[outPtr+4] = in[inPtr+4];
            total += in[inPtr+4];
            outPtr += 5;
            inPtr += 5;
        }
    } else {
        total++;
        outPtr += 4;
        inPtr += 4;
    }
}

} // end of run-of-zeros
} // end while still within scanline

```

FIG-37

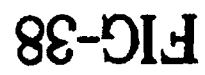


FIG-39

【図39】

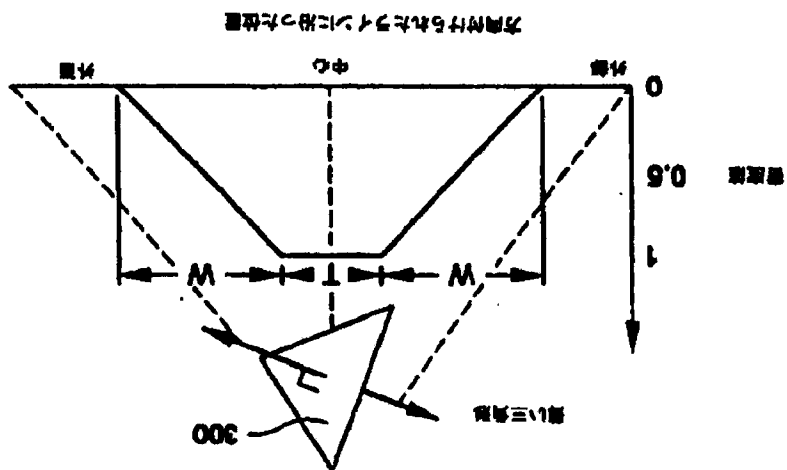
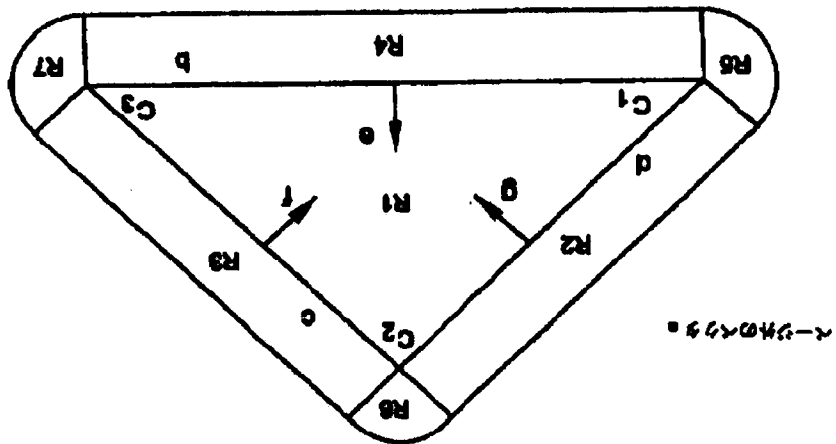


FIG-40

【図40】



```
Define Plane(A, B, C, D);  
Find triangle bounding box(bb);  
Dist = A x bb.min.x + B x bb.min.y + C x bb.min.z +  
D;  
xStep = A;  
yStep = B - A x bb.width;  
zStep = C - B x bb.height - A x bb.width;  
For z = bb.min.z to bb.max.z with unit steps  
  For y = bb.min.y to bb.max.y  
    For x = bb.min.x to bb.max.x  
      store f(Dist) in [x,y,z] (voxelize)  
      Dist = Dist + xStep;  
    end For  
    Dist = Dist + yStep;  
  end For  
  Dist = Dist + zStep;  
end For
```

FIG-41

FIG-42

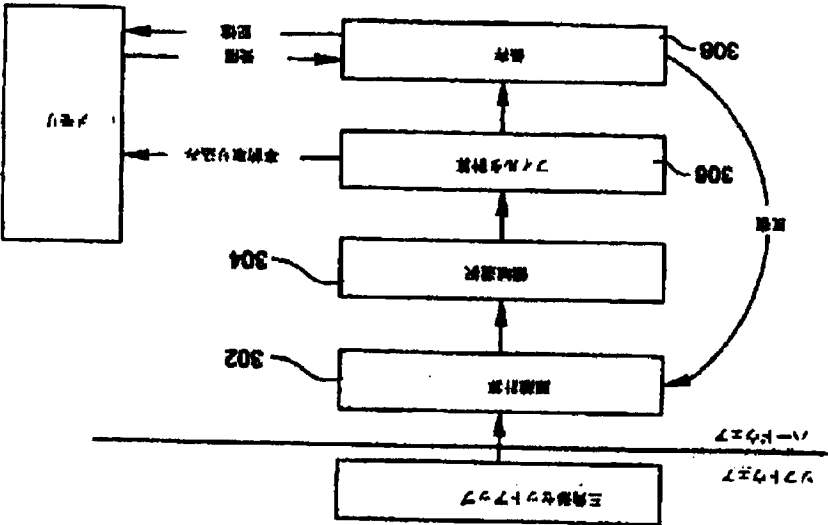
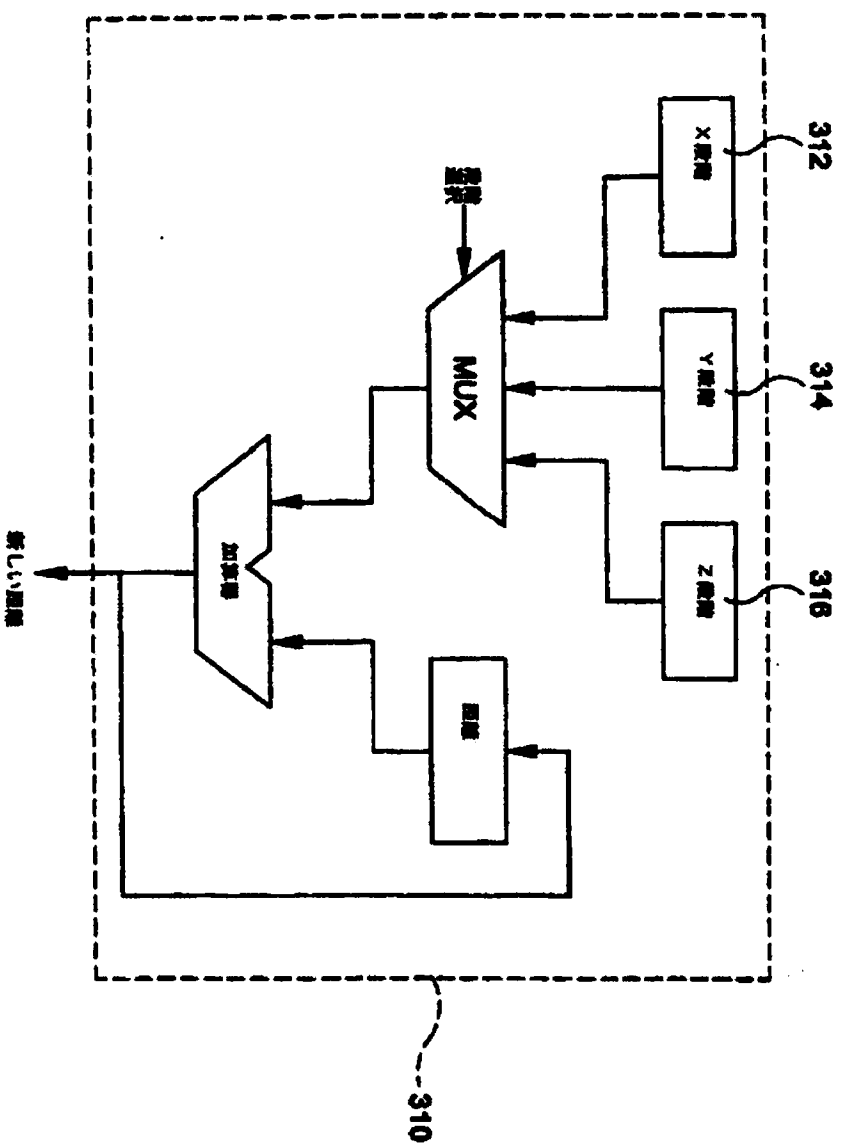




FIG-43

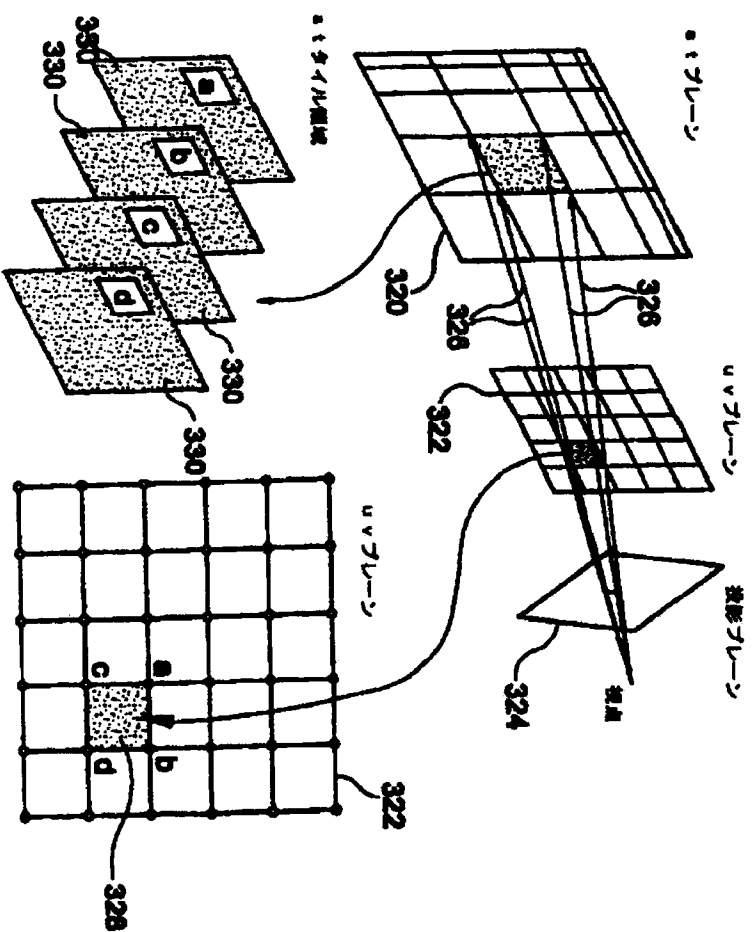


【図 43】

(161)

特表2002-520748

**FIG-44**



【44】

(162)

特表2002-520748

INTERNATIONAL SEARCH REPORT		International application No. PCT/US99/16038	
A. CLASSIFICATION OF SUBJECT MATTER			
IPC(6): G06T 17/00 US CL: 345/419, 424			
According to International Patent Classification (IPC) or to both national classification and IPC			
B. FIELDS SEARCHED			
Minimum documentation searched (classification system followed by classification symbols) U.S. : 345/418, 419, 424, 420, 425			
Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched			
Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)			
C. DOCUMENTS CONSIDERED TO BE RELEVANT			
Category*	Clonon of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.	
A	US 5,724,493 A (HOSOYA et al) 03 March 1998.	1-57	
A	US 5,760,781 A (KAUFMAN et al) 02 June 1998.	1-57	
A,P	US 5,847,711 A (KAUFMAN et al) 08 December 1998	1-57	
A,P	US 5,850,226 A (NAGASAWA et al) 15 December 1998	1-57	
A	US 5,877,779 A (GOLDBERG et al) 02 March 1999	1-57	
Further documents are listed in the continuation of Box C. <input type="checkbox"/> See patent family annex. <input type="checkbox"/>			
Special categories of cited documents: "A" document defining the general state of the art which is not considered to be of particular relevance "B" document published on or after the international filing date which may show doubts on priority claim(s) or which is cited to establish the published date of another claim or other special reason (as specified) "C" document relating to an oral disclosure, use, exhibition or other means "D" document published prior to the international filing date but later than the priority date claimed			
Date of the actual completion of the international search 12 SEPTEMBER 1999			
Date of mailing of the international search report 15 October 1999			
Name and mailing address of the ISA/US Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230		Authorized officer CLIFF M. MATHIAS Telephone No. (703) 305-9594	

フロントページの続き

(81)指定国 EP(AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OA(BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG), AP(GH, GM, K, E, LS, MW, SD, SL, SZ, UG, ZW), E, A(AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, G, E, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, M, N, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZA, ZW

(72)発明者 ビター イングラー  
アメリカ合衆国 ニューヨーク州 11790  
ストニーブルック ストニーブ  
ルック ロード 1456  
(72)発明者 チェン ボーカン  
アメリカ合衆国 ニューヨーク州 11790  
ストニーブルック チャペル コ  
ン  
フレックス 603  
(72)発明者 グシール フランク  
アメリカ合衆国 ニューヨーク州 11705  
ペイボート テラス ロード 546  
(72)発明者 クリーガー ケヴィン  
アメリカ合衆国 ニューヨーク州 11777  
ポート ジェフソン ランフォール  
アベニュー 115  
Fターム(参考) 5B057 CA01 CA08 CA12 CA16 CB01  
CB08 CB13 CB16 CD20 CH05  
CH14  
5B080 AA17 CA04 CA05 DA07 FA03  
FA08 FA17 GA04 GA22

【要約の続き】

が処理されてしまいうまでレイ・キャスティング、合併／分割の段階を繰り返す段階とを含む。本発明の装置および方法は、テクニカル・ワット・オブ・画像ペー  
ス・レンダリングを含めて、一回の画像操作で高解像度  
ボリウム・レンダリング、表面、ボリウム混合を行  
える真のリアルタイム性能を達成する。